EYE KANPUR : Gripping Polyhedral Objects Through Robot Vision

By

MUSUNUR LAXMI PRASAD

DEPARTMENT OF MECHANICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

MAY, 1991

EYE KANPUR : Gripping Polyhedral Objects Through Robot Vision

A Thesis Submitted
in Partial Fulfilment of the Requirements
for the Degree of
MASTER OF TECHNOLOGY

By
MUSUNUR LAXMI PRASAD

to the
DEPARTMENT OF MECHANICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR
MAY, 1991

CERTIFICATE

This is to certify that the work contained in the thesis entitled **EYE KANPUR : Gripping Polyhedral Objects through Robot Vision** has been carried out by Musunur Laxmi Prasad under our supervision and it has not been submitted elsewhere for a degree.

(Dr H.Hatwal)

Assistant Professor,

Dept of Mechanical Engg,

IIT Kanpur

(Dr H. Karnick)

Assistant Professor,

Dept of Computer Science & Engg,

IIT Kanpur

ABSTRACT

Name of the Student : Musunur Laxmi Prasad
Roll No. : 8910518
Degree for which submitted : M.Tech
Department : Mechanical Engineering

Thesis title : **EYE KANPUR : Gripping Polyhedral Objects through Robot Vision**

Names of the thesis supervisors :

1. Dr H Hatwal Mechanical Engineering,
2. Dr H Karnick Computer Science and Engineering.

Month and Year of submission :May 1991

This work describes how to grasp an unknown polyhedral object with the aid of robot vision. A CCD Camera image is processed to form a line diagram. The line diagrams are interpreted in the high-vision module to determine and locate the grippable surfaces. Grippability is assured if two parallel surfaces, with the requisite area, on the object are located.Two views (usually taken at 180°) or more may be required.

We assume that our objects are convex polyhedra. This leads to numerous simplifications with no inherent approximation as is usually the case with curved surfaces. Ideally an image of a polyhedron would consist of regions of uniform intensities and can be conveniently represented by a line diagram.

We interpret these line diagrams extracted from multiple views of the object , with a set of rules based on straight forward geometric constraints to locate two surfaces that are parallel in space. If found, these planes are examined to see if they satisfy all the requirements for a legal grasp configuration. Actual gripping requires the determination of both location and orientation of these planes in the world coordinate system.

We do not construct an explicit 3-D model of the object and thus we avoid any database matching.Once the location and orientation parameters are determined, co-ordinate transformations are used to determine the joint parameters for a PUMA-560 robot which was used to grip the object.

ACKNOWLEDGEMENTS

I extend my heartily thanks to my thesis supervisors Dr H Hatwal and Dr H Karnick for giving me the freedom to experiment, valuable guidance and a word of caution when ever I was getting off the rails. I am grateful to Dr A Ghosh for the all round advice he has given me from the day I joined my Master's programme.

I wish to thank Dr R N Biswas and Dr Sumana Gupta for giving me an insight into Digital Electronics and Image processing.

What ever little work I had in getting my setup ready was readily accomplished due to the constant cooperation of Mr R M Jha of Manufacturing Science Lab. I also like to thank Mr Bharthia, Mr Prem Prakash, Mr Panna Lal and other staff of the lab in this regard.

It was a pleasure to be associated with Center For Robotics for nearly two years and I did gain a lot from the discussions we have had . I would like to mention Mr Susmit Sen, Dr S R Pandian , Mr Podder, Mrs Kulkarni, Miss Sushma and Mr Vivek Shukla for their help.

I would like to thank both my seniors and juniors for their help and encouragement. I did receive a lot of help in C-programming from Mr Anupam Bagchi and Mr D Venkaiah. Praveen and Ravi Shankar were always there for troubles in Lisp. Mr Samiran Mandal helped me getting the papers etc.

What ever problems I had with Microsoft C were cleared in no time by Shastry and Srikant of CAD.

Thanks to all my friends who have directly or indirectly helped me in having an enjoyable stay at IIT Kanpur.

CONTENTS

List of Figures and Tables

CHAPTER 1 INTRODUCTION

1.1 Introduction :

Automation and Robotics :

Mass-production assembly lines were first introduced at the beginning of the twentieth century (1905) by Ford Motor Company. Over the ensuing decades, specialized machines have been developed for high volume production of parts. Here the machines and processes are often very efficient, but they have *limited flexibility*. Robots on the other hand offer *flexibility* to the manufacturing processes with which they are integrated. They achieve this flexibility because they can be reprogrammed to accommodate model variations, product changes or new operations.

In the beginning robots were used only in hot, smelly and hazardous jobs. But today they cover a wide range of applications. Some of these are loading/unloading of die casting machines, spot welding of automobile bodies, arc welding, deburring, sealing, gluing, inspection and in assembly operations. One major *disadvantage* in using robots is that expensive jigs and fixtures are required for positioning the object in the right position and orientation so that the robot can pick it up from a previously known point. This can be efficiently handled by the use of a vision system in the control loop.

In this thesis we address the problem of how a robot can grip an unknown object whose position and orientation are not known a priori.

Robot Vision :

Robot Vision studies how the visual sense modality can

be used in the control loop of robots to carry out different tasks. A vision system analyzes images and produces descriptions of what is imaged. These descriptions should capture those aspects of the objects being imaged that are useful in carrying out some task involving the objects. Thus the robot vision system can be considered as an element of a feedback loop that is concerned with sensing, while other elements are dedicated to decision making and implementing these decisions.

Robot vision systems supply valuable information that can be used in

Collision avoidance : Vision is used to detect the presence of unknown obstacles to help in the path control for collision avoidance.

Navigation of mobile robots : Servoing errors which can lead to deviation from precomputed path for mobile robots can be corrected through vision.

Identification of parts : If a mix of different varieties of parts are to be separated, then identification of the individual parts can be done using vision.

*Inspection of parts :*Vision can be used when components are to be checked for flaws or sub assemblies are to be checked for missing components etc.(e.g. **KEYSIGHT [1]**)

Vision based Manipulation :

Robot Vision can be used in an effective and economic manner to permit Industrial Robots to handle imprecisely positioned or unoriented workpieces and subassemblies to compensate for buildup of errors in tolerances. This would in effect fine-tune the position and orientation of the end-effector in picking up an object. Thus the cost of expensive jigs and fixtures, that are

other wise required to maintain the position and orientation of the object can be done away with. Moreover jigs and fixtures are only for *known objects*. But vision can be used to handle altogether *new objects* that are not stored in the database.

In the case of separated workpieces lying stably on the bed one can get an un-obstructed view of the object. In the worst case they may be touching each other. The methods developed for this case can be carefully modified to take care of objects on a moving bed with variable speed. If the workpieces are occluding each other one can introduce a mechanism to scatter the objects to give un-occluded views of the objects or use multiple views of the scene.

If the Workpieces are hung on hooks i.e. are being transported by some overhead chain conveyor or equivalent the methodology developed in the first case may not suffice. This definitely requires 3-Dimensional information to determine the position and orientation of objects in the World Co-ordinate system.

Consider a totally jumbled container having randomly positioned, interlocked workpieces. No clear un-obstructed view of any of the workpieces would be available. This could be the *most difficult* situation to handle because, apart form all other tasks we have to isolate the image region corresponding to a single object.

If the work pieces are arranged in trays and then stacked in a bin, the problem would be to distinguish the workpieces from the tray material. Although they may not be positioned precisely in the containers the permitted variations in the position would be small.

Robotic Gripping :

The design and operation of any gripper would be

governed by the nature of the workpiece being handled, strength of grip etc. There are numerous designs of grippers to cover a large range of objects. The simplest one used in industry is the pneumatically operated parallel jaw gripper. This can take care of most of the pick and place operations. Other grippers include 3-jaw, magnetic, vacuum etc. And there are some special designs of grippers that can handle eggs and bulbs.

The aim of the present work is to grasp *unknown* polyhedral objects with the aid of robot vision, using a parallel jaw gripper.

1.2 Previous work :

We briefly review earlier work under two main headings namely *Robot Vision* and *Robot Grasping.*

Complete Vision systems for a specific purpose :

One of the first projects in which visual information was used to plan the motion of an Industrial Robot was **COPY-DEMO** [2], built in 1970 by Patrick Winston , Berthold Horn and Eugene Freuder at MIT. It was a simple closed-loop hand eye system that uses a mechanical manipulator to build a copy of a structure composed of childrens toy blocks. A more sophisticated system having practical application **CONSIGHT-I** [3] was developed at GENERAL MOTORS in 1978 by Holland, Rossol and Ward. It was a Vision-controlled Robot system for transferring parts from a Belt conveyor. No occlusion was considered and the object was lying on a belt moving at a variable speed. A *one-dimensional* CCD array was used for the camera and processing was done on binary images where only the information of the boundary was made use of. **KEYSIGHT** [1] was another vision system developed by General Motors that was used in inspection. Using grey-level pixel data it

examined the spring assemblies on the engine heads for the presence of valve spring cap keys.

A vision system [4] that acquires cylindrical workpieces from bins was developed at University of Rhode Island (1982) by Kelley, Birk and Martins. They considered workpieces partially organized and unseparated.

(a) Robot vision :

Robot Vision is a vast area dealing with a variety of topics. We present a brief survey of work done in the areas of *Edge detection, Detection of corners, Determination of orientation, shape and location*

(i) Edge Detection :

Edge detection has been one of the most active fields in vision. The classic paper by **Roberts[1965]** showed how to extract edges from photographs of polyhedral objects digitized on a covered drum plotter. Duda & Hart [1972] considered use of the *Hough Transform*, which is a mapping into the parameter space of the line sought after. The results produced by the simple edge finding techniques left a lot to be desired. Griffith [1973], for example, introduced assumptions about the statistics of image features, while Shirai [1973] exploited expectations about the scene to focus attention on areas where lines were likely to be found. Hueckel [1971,1973] developed a least square fitting scheme based on the expansion of the brightness function in a region in terms of orthogonal functions. Application of the first derivative operator is followed by a search for peak values. A method to suppress large values near the local maxima is also required. Partly for this reason, second derivative operator became popular, since the edge is located where their output is zero. Linear first-derivative operators are directional but second-derivative

operators can be made rotationally symmetric.

Horn [1972] showed that laplacian was the lowest order linear operator that preserved all information required to reconstruct an image. The *Gaussian* is popular for smoothing operations for several reasons, first it is a rotationally symmetric operator that is the product of two 1-D operators,[Horn 1972]. And second it has the smallest product of width in the spatial and frequency domains.

Marr [1976] at first proposed a directional operator for edge detection. He later claimed that a rotational symmetric operator was optimal. The contours where output of the operator passes through zero are called *Zero Crossings*. A new theory of edge detection based on finding the zero-crossings of the output of an even-derivative operator applied to the image is described by Marr and Hildreth [1980]. They took the laplacian operator and combined it with a Gaussian smoothing filter to produce a rotationally symmetric edge operator. Ideally the zero-crossings of the filtered output are closed contours.

Canny [1983] has developed a one-dimensional operator that provides an optimal trade-off between localization and detection. He designed his operator to minimize the sum of two error criteria, namely the probability of failing to mark an edge or announcing an edge where there is none, and the distance between the detected edge and the true edge. His first error criterion also implicitly forces the probability of obtaining more than one response to a single edge to be low. Canny restricted his search for a solution to the space of linear, shift-invariant operators. He confirmed the trade-off between signal-to-noise ratio and localization and showed that the optimal operator can be written as a sum of four complex exponentials. The result contains an arbitrary spatial scale factor, so that operators of different

sizes can be designed.

This is still an active area as indicated by the recent paper by Hartley [1985].

(ii) Detection of Corners :

The earliest corner detection methods involve first segmenting the image into regions and representing that object boundary as a chain code. Corners were identified where the direction changed rapidly [5]. Later attempts were directed at coming up with a corner detector which operated directly on gray level images. These include the one developed by Zuniga and Haralick [6], Kitchen and Rosenfeld [7], one by Dreschler and Nagel [8] and the one by Rangarajan et al [9]. Rangarajan et al formulated it as an optimization problem. They modeled the local gray level function around a corner point with additive Gaussian noise and attempted to find an optimal function representing the corner detector which when convolved with the gray level function yields a maxima at the corner point.

(iii) Object location, Orientation and Shape :

Image understanding research has produced various techniques for extracting information about visible surfaces of a scene. Early work was done by many researchers.

Yuan [10] presented a method for determining the 3-D position and orientation of an object relative to a camera based on a 2-D image of known feature points located in the object. Though this problem is solved in classical photogrammetry making use of the collinearity condition, Yuan arrives at a solution by exploiting the algebraic structure of the problem, independent of the configuration or number of feature points.

Liu & Faugeras [11] present a new method for

determination of camera location from 2-D to 3-D straight line or point correspondences i.e. known feature or control points are made use of in this process.

Horn and Ikeuchi [12] have presented a system that locates and grasps parts from a pile. Using photometric stereo to make surface orientation measurements the camera field is segmented into isolated regions of a continuous smooth surface. One of these regions is selected as the target. *Attitude* of the physical object associated with the target region is determined by histogramming surface orientations over that region and comparing them with histograms obtained from prototypical objects. Range information not available from photometric stereo is obtained by the PRISM binocular system. A collision-free grasp configuration is computed and executed using altitude and range data.

Sandini et al [13] have designed a system for extraction of superficial features from a sequence of range data. It is done using depth maps because they may constitute a common representation for many different processes such as depth from stereo, depth from occluding contours, range finders etc and hence facilitate data integration for a complete description of the object imaged. Some other works in that direction are Shape from Shading [Horn][14,15], Surface structure and 3-D motion from Image flow Kinematics [Waxman, Ullman][16], Structure from Motion [Ullman] etc.

For computation of depth Faugeras and Heber [17] have tried a method that uses a combination of what is known as *Active Triangulation and Time of Flight*. Active triangulation uses an extra source of light to project some pattern onto the objects to be measured thereby reducing the complexity of *Stereo matching*.

1.3 Grasping :

The problem of Grasping has drawn much attention in recent years. Hanafusa and Asada [18] presented a 2-D analysis in which a potential function based on the shape of the object is used to determine stable positions for the placement of a pointed, frictionless finger. Okada [19,20] derived equations of motion for a sphere and rectangular box manipulated by these fingers with hemi-spherical tips. Salisbury and Craig [21] develop a more general 3-D analysis in which a jacobian relates forces and velocities of the fingers to an equivalent force and velocity of the object.

Kerr and Roth [22] deal with determination of finger joint motion to produce a desired motion of the object and determination of the workspace of the hand for special cases of planar and spatial hands. Randy C Brost [23] investigated automatic grasp planning in the presence of uncertainty. The algorithm he presented deals with automatic planning of robot gripping motion that is insensitive to bounded uncertainties in the object location. Uncertainity problem can be removed if vision is used in conjunction.

1.4 Objective of the present work :

The aim of the present work is to grasp unknown polyhedral objects with the aid of robot vision. First, the view taken through a CCD Camera is processed (at PC-AT) to form a line diagram. The line diagrams are interpreted in the Hi-vision module to determine and locate the grippable surfaces.

We assume that our objects are convex polyhedra. This leads to numerous simplifications with no inherent approximation as is usually the case with curved surfaces. Ideally an image of a polyhedron would consist of regions of uniform intensities and can

be conveniently represented by a line drawing.

We interpret these line drawings in the Hi-vision module (workstation) according to a set of rules to look for the existence of two surfaces that are parallel in space. Once this is done ,these planes are examined to see if they satisfy all the requirements for a legal grasp configuration. Actual gripping will require the determination of location and orientation of these planes in the world coordinate system. This process of searching for parallel grippable planes needs more than one view. The first attempt is made through two views taken at 180^{o} apart about the vertical line.

We do not construct an explicit 3-D model of the object. We try to analyze the given object for grippability based on a set of rules and thus avoid any database matching.

Once the location and orientation parameters are determined, co-ordinate transformations are used to determine the joint parameters for a PUMA-560 robot which was used to grip the object.

1.5 Organization of thesis:

In chapter 2 the Schematic diagram of the whole vision system **EYE KANPUR** is presented with a brief description of the hardware detail of machines/equipment involved in the Low-vision module. Also details of implementation of the Low-vision module are presented.Chapter 3 outlines the Hi-vision module of the system where the details of rules implemented together with the Camera transformation to map the camera coordinates to the World Coordinate System are discussed. The validity and applications of the results are discussed in Chapter 4. Conclusions and directions for future work constitute Chapter 5. Appendices deal with the specifications of the equipment (Appendix A for camera and Appendix B for frame grabber).

CHAPTER 2 LOW-VISION MODULE

2.1 Introduction :

A schematic of the hardware layout has been shown in Fig 2.1. We briefly describe the methodology followed and the hardware requirements. The vision system *EYE KANPUR* can be broadly classified into Low-vision and Hi-vision modules. The Low-vision module essentially does image processing and finally produces a line diagram of the object imaged. The line diagram is sufficient to represent the image of a convex polyhedron. The Hi-vision module analyzes this line diagram to ascertain the grippability and then determines the position and orientation required by the robot to grasp the object. The Low-vision module of Fig 2.1 is discussed in this chapter and the Hi-vision module is explained in next chapter.

2.2 Hardware components of Low-vision module :

i) A CCD Camera for the imaging.

ii) A Frame Grabber to digitize and store these images in a frame memory

iii) IBM PC-AT for housing the Frame Grabber.

iv) ETHERNET card for the communication between PC-AT and Workstation through ETHERNET.

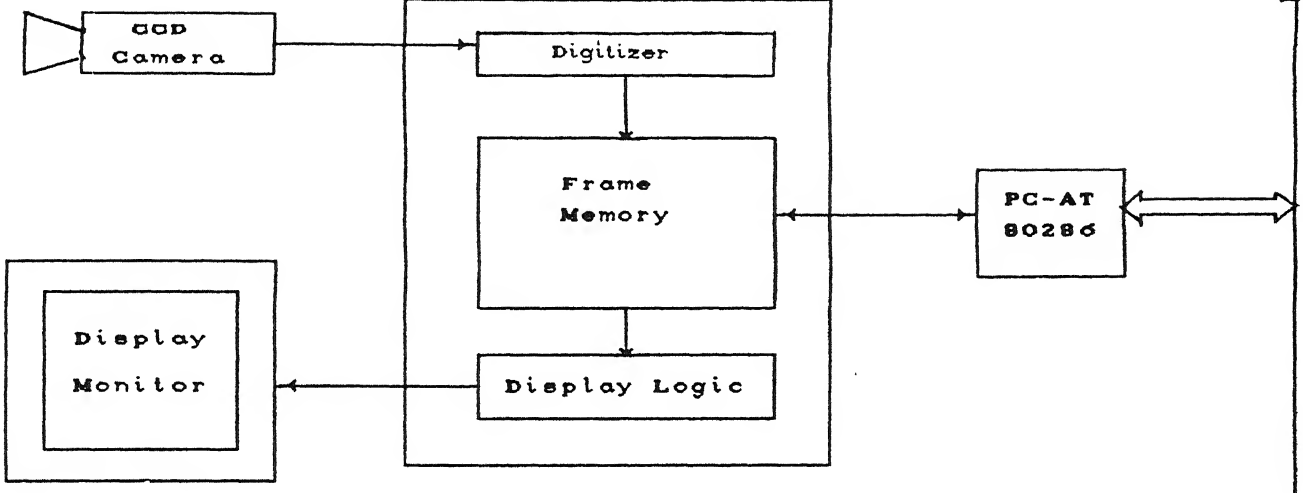
2.2.1 *Camera*

This consists of PULNiX TM-560 camera with an attached lens system. The imager is 8.8(mm) X 6.6(mm)and the CCD array is of size 500(H)X 582(V).

The resolution in Horizontal direction equal to

$$\frac{8.8}{500} \quad \text{i.e. } 0.0176 \text{ mm}$$

The resolution in vertical direction is



Low-Vision Module :

1. Capture an Image
2. Detect Edges
3. Thinning
4. Segmentation & Line Detection
5. Formation of LINE DIAGRAM

Hi-Vision Module :

1. Formation of Valid Visual planes
2. Ascertain the grippability
3. Determine the position and the orientation (**O A T**) for PUMA to grip the object

Work Station (68030) 25 MHz 8 Mb

E T H E R N E T

FIG 2.1 EYE KANPUR HARDWARE CONFIGURATION

$\frac{6.6}{582}$ i.e. 0.0113 mm.

These values are used in converting the pixel units to mm be sed in camera transformation.

Other specifications are given in Appendix A

2.2.2 *Frame Grabber :*

The PCVISIONplus Frame Grabber (Imaging Technology,USA) s a video digitizer and frame memory capable of digitizing standard RS-170/330 or CCIR (TM 560 gives a CCIR output) video input and storing the digitized image in a special on-board frame nemory. The image can be simultaneously displayed on video nonitor.

The PCVISIONplus is a single board that plugs directly into an expansion slot in the IBM PC. It digitizes the incoming video signal to eight bit accuracy at a rate of 30 frames per second and stores the resulting pixels in frame memory. Each pixel is one of 256 gray levels. Two 512 X 512 or One 640 X 512 image can be stored. Display logic converts the pixels in the frame memory back to analog RS-170 format for display on a video monitor. There are three output channels for pseudo color display.The one input and three output channelseach contains eight look-up tables (LUTs). Each of these 32 tables contains256 entries. They are helpful in performing transformation of the 256 intensity levels. i.e. simple point transformations can be done without any processing or delay.

Further specifications are listed in Appendix B.

Software for the PCVISIONplus is supplied as a callable C library of image processing routines called **ITEX library.**

2.2.3 *IBM PC-AT*

An IBM PC-AT or 100% compatible with following specification is needed for housing the frame grabber :

8-bit data bus,24-bit address bus that allows mapping into extended address space.

16 I/O mapped control registers mappable to any 16-byte boundary.

2.2.4 *PCNFS*

PCNFS is the PC version of Sun Microsystem's Network File System. It allows sharing of files between users of different operating systems in the NFS network including Unix and VMS.

Essentially it gives us the flexibility of using either of the machines without being bothered by DOSs' limitations. Thus we have a general platform over which one can implement any algorithm in robot vision. Particularly a 68030 based workstation for Hi-vision module gives the much desired speed.

2.3 Software developed for Low-vision module :

In this section we present the algorithms that we have used. The Low-vision module takes in gray level image of the object and gives line diagram as output. The various steps involved are

Image acquisition
Edge detection
Thinning
Extraction of Contours
Reset Contour
Extraction of lines
Line Joining

We assume that the object is a convex polyhedron and is the nly object in the scene.

2.3. *Image acquisition :*

Image acquisition is done by the library routine *grab* to :ontinuously grab the image and display it on the monitor. It can)e frozen at a chosen instant by *snap* to store the image in the 'rame memory. Now we have the image in 512 X 512 array i.e. the 'rame memory.

2.3.2 *Edge detection :*

We have used *sobel's edge operator*, from the **ITEX** library to find the edges for the following reasons :

.It is less sensitive to noise than a second-derivative operator.

.Works better than other gradient operators like *Roberts* and gives finer and continuous edges (due to the size of the kernel being 3X3)

.It can be implemented readily in any hardware and if we can get a hardware scroll window of size 3X3 we can implement 3X3 kernel with ease and obtain satisfactory results.

The *Scaling factor* would scale the output of the sobel operator. Scaling is done by dividing the pixel value with this factor. A more detailed discussion on the effect of scaling factor is presented in chapter 4. We observed that a factor between 40 and 60 was performing well for the present lighting conditions.

2.3.3 *Thinning :*

In thinning all that we are now doing is to replace a

horizontal wall of thickness t1 starting at s1 by a point at s1+t1/2 at same y. This is illustrated in Fig 2.2 (Image coordinates are explained in *Chapter 4* , Fig 4.1).

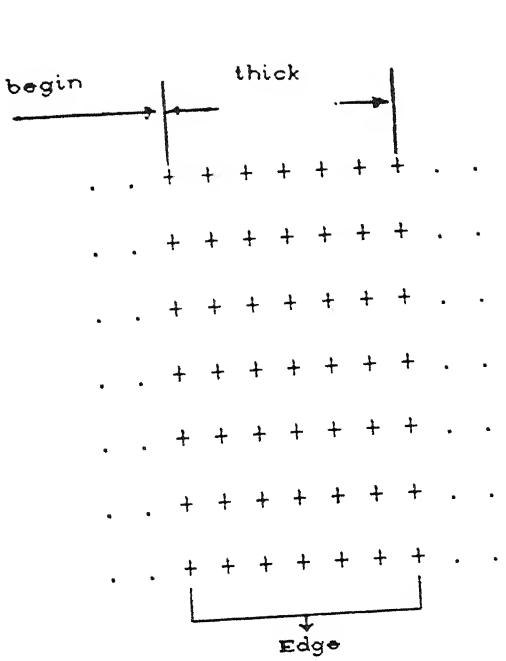
Algorithm :

```
For y=0 to 512
Read a horizontal line (512) pixels into a  linbuf[512]
For x=0 to 512
if linbuf[x]=0 & linbuf[x+1]>0
     {
          thick=1 /* wall thickness */
          begin=x /* beginning of wall */
      }
if linbuf[x]>0 & linbuf[x+1]>0 thick++;
if linbuf[x]>0 & linbuf[x+1]=0
     {
     if thick=1
          wpixel(begin,y,200); (write into a pixel)
     else
          point=begin+ thick/2
          wpixel(point,y,200);
     }
```

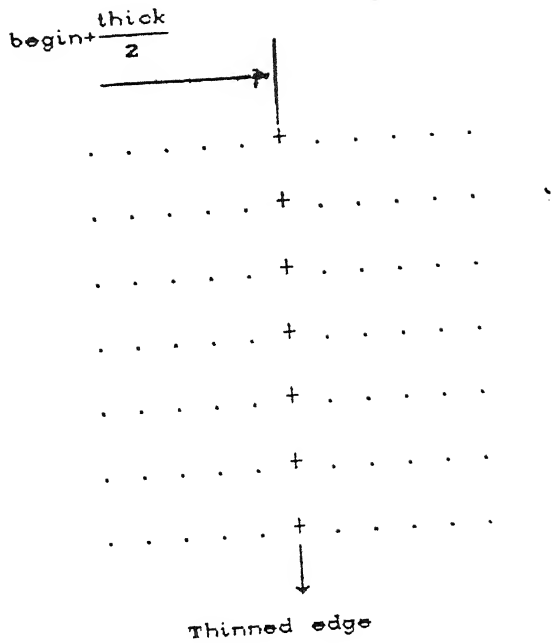
Thus this can thin down edges to a single pixel width.A strategy that identifies the blob (*where a corner of the polyhedron exists*) and handles it in a different manner could give us a better skeleton that is closer to the original image. More details on desired improvements in thinning are given as suggestions for future work in chapter 5.

2.3.4 *Contour Finder* :

This module is called *SEGMENT* in the code . *SEGMENT* uses a SEGMENT_THRESHOLD which is the allowable deviation for a pixel to lie on a contour. i.e. from the current pixel another pixel within



a) Input



b) Output

Fig 2.2 Thinning

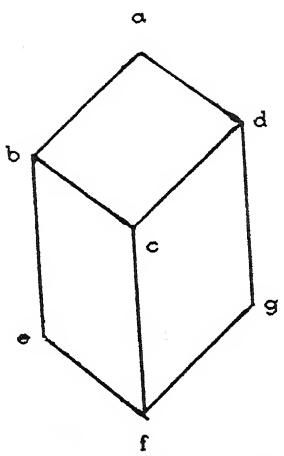


Fig 2.3 (a) Input for the segmentation module
(a thinned image)

his deviation will be picked up to form the next point in the ontour. If there is more than one pixel falling with in this hreshold then the nearest pixel is chosen. Thus in a top-left to ottom-right scan one of the contour is detected. If the input i.e. the thinned image) is of the form shown in Fig 2.3 (a) then ig 2.3 (b) shows a *possible* sequence of contours to be detected hen *SEGMENT* is called repeatedly. The sequence of contours etected may be different when an actual digital input is given. xperimental details are given in chapter 4.

Algorithm :

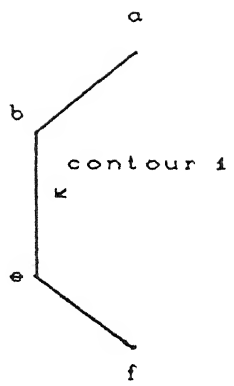
```
for j=0 to 511 (y-direction)
for i=0 to 511 (x-direction)
get hold of a lighted pixel and then mark it as x1,y1:
(we are interested in just one)
if(any other lighted pixel falls within THRESHOLD to (x1,y1))
mark new point for reference
else next j;
```

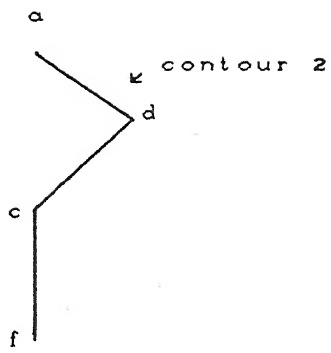
There are some more implementational details that are not presented here but can be found in the comments along with the code.*Globaly* is a restart point for this entire process so that it would not start all afresh from the top.

2.3.5 *Reset Contour :*

Here pixels found in the contour are switched off. This helps in getting the *end condition* for processing the picture. When there is no other lighted pixel the cycle consisting of

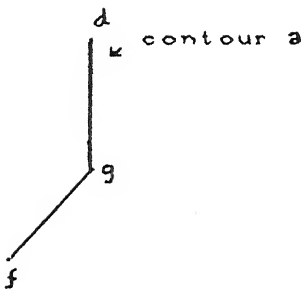
Detect Contour
Reset Contour
Line Finder terminates.

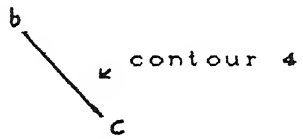




First Scan *Second Scan*

Fig 2.3 (b) Possible output of segmentation module
After repeated scans





Third Scan *Fourth Scan*

2.3.6 *Line finder :*

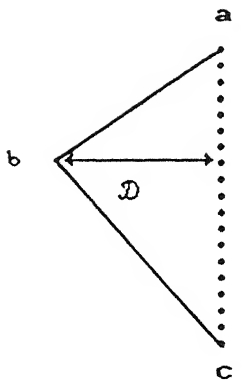
This module is called *NEW_SEPARATE in the code*. It uses a EW_SEPERATE_THRESHOLD which is the allowable deviation for any ixel to lie on a digital straight line. It is illustrated in Fig .4 (a) where *b* can not become a member of the digital straight ine from *a* to *c* because it exceeds this threshold. Moreover *b* has he maximum deviation from the line joining *a c* . Hence *b* is arked as a corner in the contour *a b c*. In a contour (given out y *SEGMENT*) scanning is begun from the top point. The point aving maximum deviation from the line joining top and bottom oints of the contour (and deviation greater than IEW_SEPERATE_THRESHOLD) is selected to be the corner. This rocess is recursively done for the two parts above and below this corner in the entire contour.

Thus given a contour as input as shown in Fig 2.4 (b), the algorithm separates the valid line segments as shown in Fig 2.4 (c) by finding all the corners in the first and then joining them.

2.3.7 *Line Joining :*

Once all the lines in the given image of a polyhedral object have been found the line diagram of the object has to be formed. This module is named *OPTICAL_FLOW* in the code.

When a line diagram of an image of a polyhedral object is to be formed one has to see that lines that intersect in space emanate from the same corner in the image. This is necessitated due to the working of the *Contour_detector & Line Finder* that uniquely assign any corner to just one line. So, this module finds the intersection points of each set (all those intersecting at that corner in space) and all the lines are updated with the



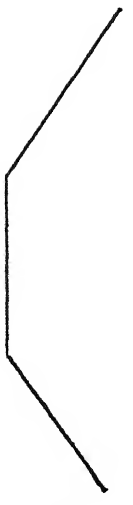
$\mathcal{D}$ - deviation

if $\mathcal{D}$ > NEW_SEPERATE_THRESHOLD

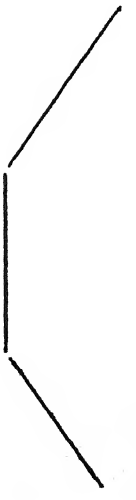
since b has the max deviation

it becomes a corner

Fig 2.4 (a)



b) input-contour



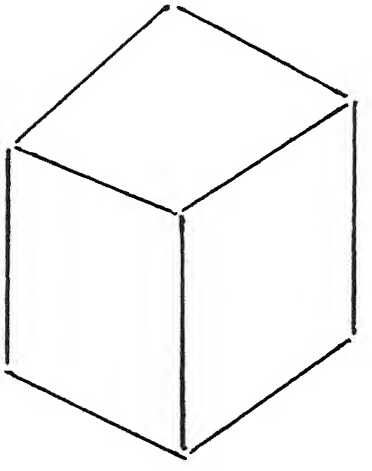
c) output- lines

Fig 2.4 Line Finder

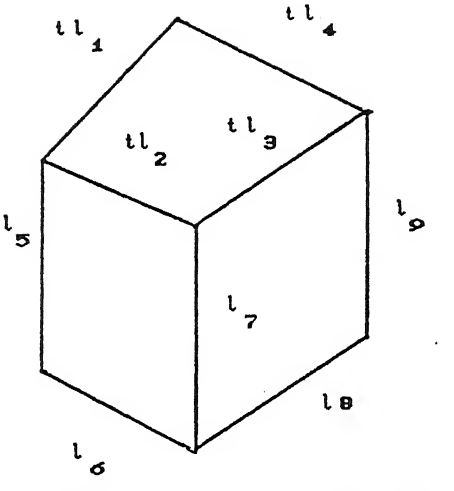
corresponding new points. Fig 2.5 (a) shows the output of Line finder (i.e. input to the line joining module). The output of the line joining module is shown in Fig 2.5 (b).

2.3.8 *Labeling* :

When one view is to be correlated to another view, we need to know the correspondence between the lines (or points in the image)of two views. At present we are solving this correspondence problem interactively by labeling the lines.(i.e. the labeling is done by the user) That is once a line diagram of an object is formed the labels for each line are determined interactively and stored along with the line.When another view is taken the labeling process should be done consistently i.e. the same line should be given the same label.These labels are used by the *Hi-Vision* module to ascertain grippability and also in the final determination of the position and orientation of the object to be picked. Fig 2.5 (b) shows a labeled line diagram of a cube.



a) input :disjoint lines



b) output : Joined lines & labeled

Fig 2.5 Line Joining and Labeling

CHAPTER 3 HIGH-VISION MODULE

3.1 Introduction :

In this chapter we deal with the analysis or interpretation of the *line diagram* obtained from the low-vision module. The implemented algorithm is presented in detail. Since we are looking for the existence of grippable surfaces without identifying the shape completely, the presented algorithms are much simpler than those which require matching against a database of models of objects [12].

3.2 TERMINOLOGY :

Although the terms and abbreviations used are either standard or explained in the body of the chapter, they are listed here for convenience.

2-D	*Two dimensional*
3-D	*Three dimensional*
back track	*to recede in the search process*
badp	*Bad predicate*
blind alley, leaf	*a node that does not have any successors*
cdr	*in lisp this returns the given list except the first element, i.e. cdr (a b)=(b)*
node	*any point under consideration*
grand parent node	*predecessor of a parent node*
parent node	*immediate predecessor of a node*
root node	*starting node of the search*
successor node	*node that emanates from the given node*
pop	*for a list 1st element is removed stack i.e. removes the top most element*
temp plane	*Temporary storing location for plane*

	that is being formed
WCS	*World coordinate system (cartesian)*
visited nodes	*list of nodes that have been traversed*
SO	*sin* O
CO	*cos* O
SA	*sin* A
CA	*cos* A
ST	*sin* T
CT	*cos* T
Sθ	*sin θ*
Cθ	*cos θ*
Sφ	*sin φ*
Cφ	*cos φ*

3.3 Definitions :

In chapter 2 the final line diagram formed is represented as a *DCEL*. The information in this *DCEL* is used as an input for the further processing.

DCEL (Doubly Connected Edge List):

It is a list of edges. Each edge of the line diagram is represented as a list of it's associated label, slope, starting and end points.

Fig 3.1(a) shows a line diagram and it's associated *DCEL*.

Fig 3.1(b) shows the associated graph of the line diagram, if P_2 is arbitrarily chosen as the starting node. All the following definitions are explained with reference to this graph.

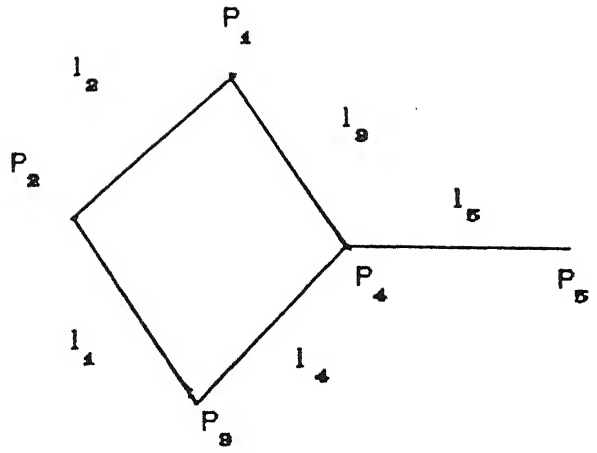


Fig 3.1 (a)

A line Diagram

Associated DCEL :if P_1 is (x1 y1), P_2 is (x2 y2), P_3 is (x3 y3) and P_4 is (x4 y4)

((l1 m1 P2 P3) (l2 m2 P2 P1) (l3 m3 P1 P4) (l4 m4 P3 P4)(l5 m5 P4 P5))

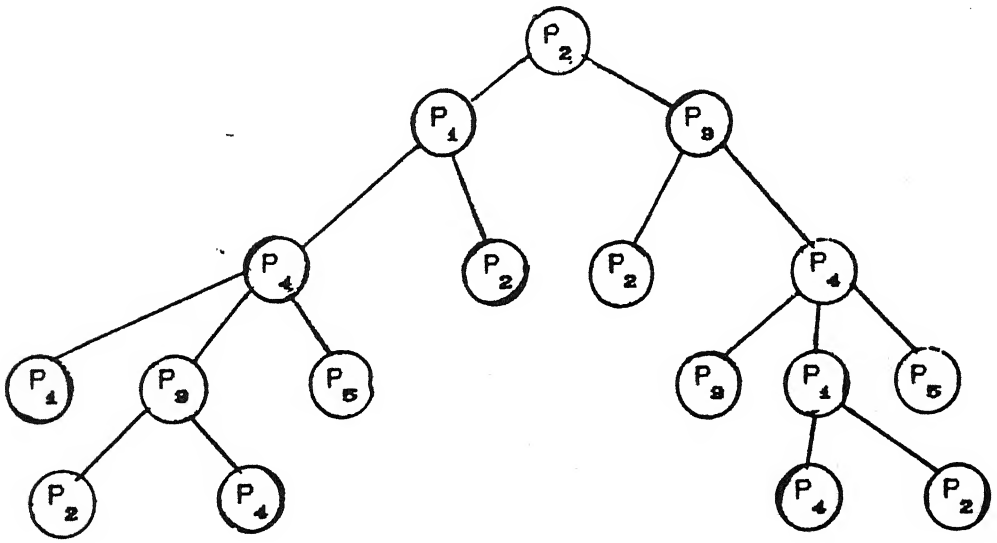


Fig 3.1 (b) Associated Graph for Fig 3.1 (a)

Successor node :

It is the node that emanates from a given node. In Fig 3.1(b) at the beginning of the graph P_2 has P_1 and P_3 as its successors.

Goalp (Goal Predicate) :

In graph search a condition that prevents further search for successors at a given node is called a goal predicate.

Visual Plane :

Any apparent plane, which may be a likely candidate for a plane in WCS is known as VISUAL PLANE. Fig 3.2(b) shows the visual planes for the line diagram in Fig 3.2(a).

Valid Visual Plane :

Visual planes that correspond to actual 3-Dimensional planes are called VALID VISUAL PLANES. Fig 3.2(c) shows this set of planes for the line diagram in Fig 3.2(a).

3.4 PROPOSITIONS :

1. A *visual plane* is said to have formed if circular connectivity of a chosen node is achieved.
2. A visual plane thus formed is *valid* if circular connectivity is achieved through minimum number lines.
3. Two distinct *valid* visual planes of an image of a polyhedral object cannot have more than one line in common. i.e.intersection of two planes in space is along a unique line.

3.5 Algorithms :

The High-vision module has been implemented in Common Lisp on a MC68030 based workstation running Unix V.3.2 . A PUMA-560 robot arm is used to do the actual gripping.

The line drawing is a 2-D projection of 3-D surfaces. We isolate these 3-D planes and then examine these planes to see if they form grippable surfaces. Once these are determined one has to determine the position and orientation of these planes in WCS. Then the corresponding joint variables for PUMA are determined so that it can REACH OUT and GRIP the object.

The main steps in the algorithm are as follows :

i) Formation of valid visual planes
ii) Search for grippable surfaces
iii) Determination of position and orientation of the Grippable surfaces
iv) Determination of joint variables

3.6 Formation of Valid Visual Planes:

An algorithm is presented for the formation of valid visual planes of an image of a CONVEX POLYHEDRAL object. The *do it yourself* methodology employed in the field of Artificial Intelligence seemed to offer potential advantages **[ULLMAN,1979]**. This algorithm is a *depth first search* with goalp defined as a set of conditions. The search at any node is abandoned if goalp is true. When the entire graph is searched, all the visual planes are formed. These visual planes are filtered using propositions 2 and 3 to form all the valid visual planes. The method helps in finding the interrelationships of 2-D planar projections of a 3-D object which might be totally unknown to the system.

The working of this module is explained with the help of Fig 3.2 (a) (b) and (c).(but the actual results are given in terms of lists involving the labels and nodes)

If the input *DCEL* to this module is corresponding to the Fig

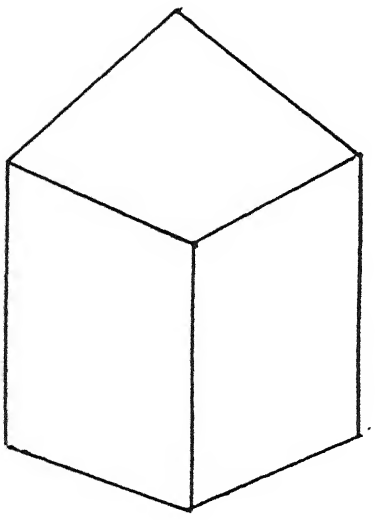


Fig 3.2 (a) Input - A line diagram

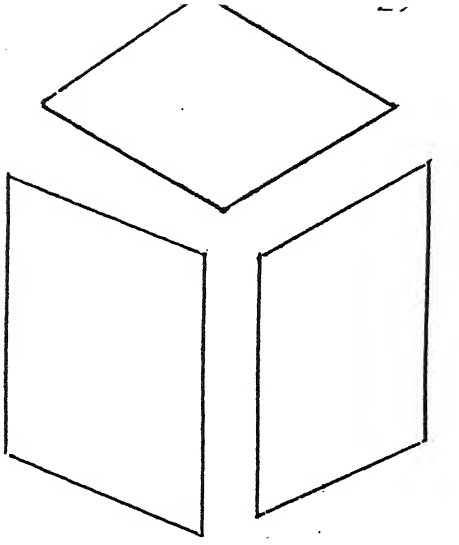


Fig 3.2 (c)

VALID visual planes

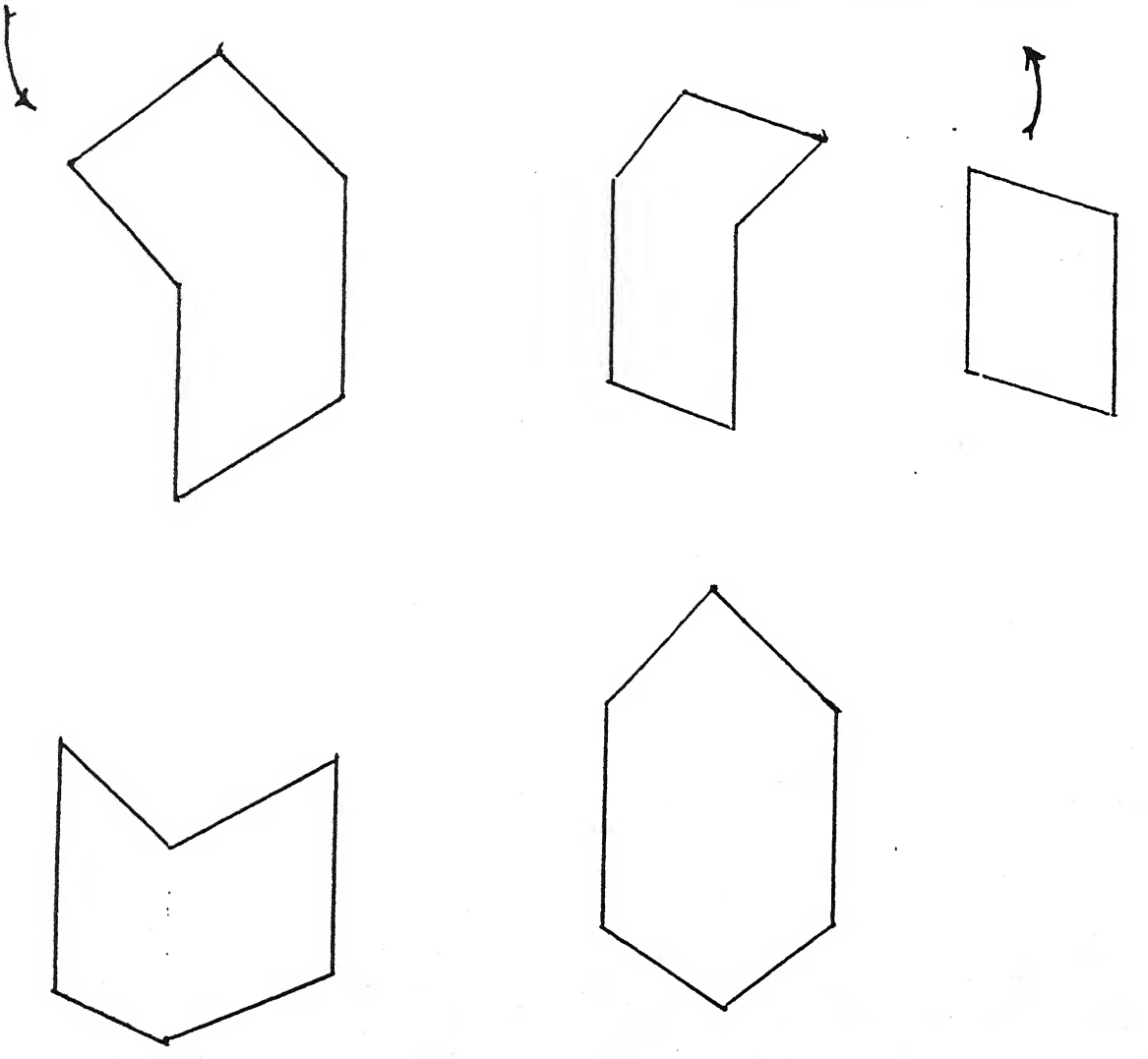


Fig 3.2 (b) Some of the visual planes formed out of the line diagram shown

3.2 (a) then first the visual planes are formed. Some such possible visual planes are shown in Fig 3.2 (b). These visual planes are then examined to give out the valid visual planes as shown in Fig 3.2 (c).

ASSUMPTIONS :

1. Objects are convex polyhedra.
2. The scene has only one object.
3. Every line in the image corresponds to an actual physical edge of the object (i.e. there are no shadows) and has a representation as a doubly connected edge.
4. The edges are labelled absolutely. This implies that an edge occurring in two or more distinct views has the same label and finding correspondences becomes trivial.

GOALP CONDITIONS :

1. When a successor node equals its grand parent node.
2. When a successor node equals the root node.
3. When a successor node equals any of the visited nodes.
4. When a node is a blind alley,i.e. it does not have any successors.

ALGORITHM :

```
    Phase 1 : Formation of all visual planes
    input is DCEL and output is VISUAL PLANES
    begin
    planes ← nil
    choose n (any node)
  function findcycle (n : node,temp_planes : list of lines)
    mark n;
    S ← non-parent successor (n)
```

```
∀  s ∈ S
if (marked (s)
{
add line (n,s) to temp_planes;
find cycle in temp_planes and add it to planes;
remove line (n,s) from temp_planes;
}
else
add (n,s) to temp_planes;
find-cycle (s,temp_plane);
end;
```

Phase 2 :Formation of valid visual plane set (VVPS)

```
 input is VISUAL PLANES (VP)
output is VALID VISUAL PLANE SET (VVPS)

begin
VVPS ← NIL
while VP ≠ NIL
do
{
VVP ← smallest cycle in VP
delete all C ∈ VP ∋ C ∩ VVP > 1
put VVP into VVPS
}
end;
```


3.7 Grippable surfaces:

A *Grasp Configuration* **[13]** should satisfy the following two conditions.

1.Given the gripper's shape and object's shape, it should produce a mechanically stable grasp. Such configuration will be

called *legal grasp* configurations.

2. Such a configuration must be achievable without collisions with other objects. Grasp configurations are limited by the relationship between the shape of the gripper and the shape of the neighboring obstacles. Such configurations will be called *collision free* configurations.

Legal Grasp Configuration :

Several definitions have been given by Hanafusa & Asada, Brady, Boissonnat etc. In the current work we are restricting our selves to finding the *Legal Grasp Configurations* as given by Lozano-Perez [24,25].

Legal grasping configuration is the one in which the object satisfies the following two conditions.

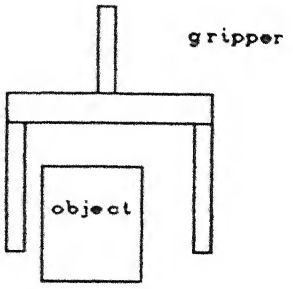
1. The object is not translated while the gripper is grasping the object

2. The object is not rotated while the gripper is grasping the object.

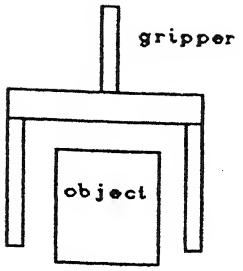
Legal grasp configuration can be achieved in following steps.

1. If the centre of area of the object does not coincide with the mid point of the two jaws of the gripper, then the simultaneous closure of the jaws would result in translation of the object. This is illustrated in Fig 3.3 (a) [i].

If the gripper is aligned in such a way that the centre of area of the object coincides with the mid point of jaws then the simultaneous closure of the jaws would not result in any translation.This is illustrated in Fig 3.3 (a) [ii].

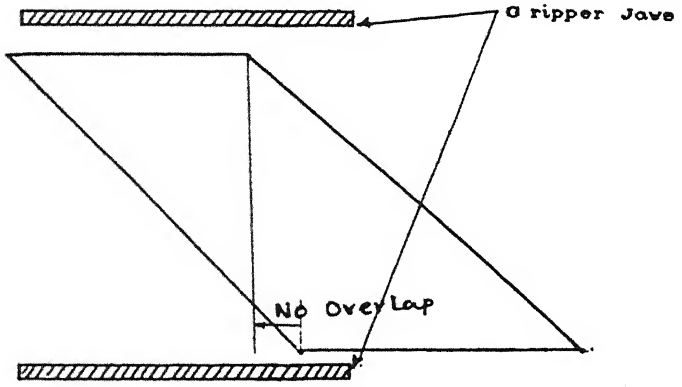


(i) Case that results in Translation

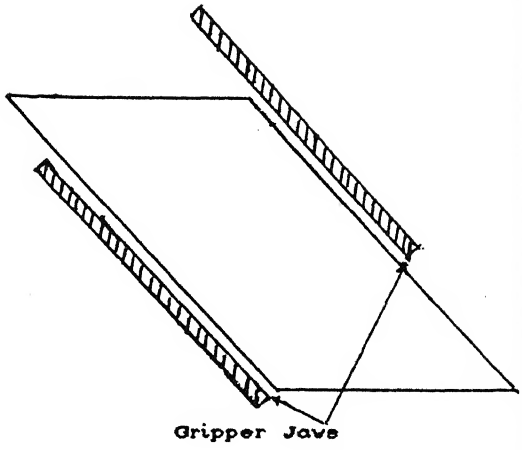


(ii) No Translation

Fig 3.3 (a)



i) Case that results in rotation of the object



ii) No rotation results because of the overlap

Fig 3.3 (b)

2. For preventing the object's rotation check for the lateral off-set between the two planes must be done to ensure that the planes have a overlap equivalent to the minimum grip area required.

In Fig 3.3 (b)[i] there is no overlap between the planes to be gripped and would result in rotation of the object.

In Fig 3.3 (b)[ii] the same object is shown but the gripping is being done along the other set of planes. This would not result in rotation of the object since there is an overlap between the planes to be gripped.

These can only be implemented after the coordinates in the WCS are known. In our work we are only calculating the center of area and orientation so that the gripper may be aligned without translating the object.

To determine a set of grippable surfaces, first one has to arrive at two parallel surfaces.

Parallelism between surfaces :

Two planes are parallel if there exists two lines (not mutually parallel) that are parallel between them.

Broadly the method of establishing the parallelism between two planes $\mathbf{plane}_1, \mathbf{plane}_2$ is as follows.

let $\mathbf{plane}_1 = (l_1, l_2, l_3, \ldots, l_p)$ consist of p line segments

and $\mathbf{plane}_2 = (m_1, m_2, m_3 \ldots m_q)$ consist of q line segments

let count=0

for i= 1 to p

for j= 1 to q

if there exists l_i such that $l_i \parallel m_j$

{ The parallelism between two planes is known by checking up the plines which is the list of sets of parallel lines.

These sets are made in the following manner :
In each view after the valid planes are formed the parallel sets are formed with in the planes, by looking at their slopes. When a line is common to two planes the relationship gets extended and in a similar way across the views }

count=count+1;
remove all lines parallel to l_i from $plane_1$ and
remove all lines parallel to m_j from $plane_2$
else next j
if count=2 then the planes are parallel
return
else next i
return nil (planes are not parallel)

In the actual implementation this was achieved in two stages.

First, only those set of planes that emanate from two parallel lines in another plane are considered. Next checking is done to see if there exists a different line that is parallel between them.

Information from a different view of the object :

When the system fails to come through the grippability analysis in one view it is essential that the information available in the second view be correlated with that of the previous one. The data-structure keeps the relationship in terms of parallelisms between the lines only, since this relationship can be extended to any arbitrary view. But this is also the short coming of our approach since we can not assure completeness of the

algorithm. Because, if a line is not parallel to any other line in one view and if this is not visible in the next view it can not be correlated to any other line. i.e. the information about the polyhedron is not complete. This can be taken care of by using the absolute orientation of the edges in space. Which means depth information for all those points would be required, hence would be a computation intensive operation.

3.8 Object Position and Orientation in WCS :

Once the algorithm determines the grippable surfaces, the robot has to reach out and grip the object. Therefore the location and orientation of the central plane (between the grippable surfaces) has to be transformed from the camera image plane to the WCS. This transformation procedure is explained below.

(i) Homogeneous coordinates :

For any world point (X,Y,Z) the homogeneous coordinates are defined as (kX,kY,kZ,k), where k is an arbitrary non-zero constant. In the vector form

$$\text{if} \quad W = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

then the homogeneous coordinates can be written as

$$W_h = \begin{bmatrix} kX \\ kY \\ kZ \\ k \end{bmatrix} \quad ..(3.1)$$

(ii) Perspective transformation :

Fig 3.4 (a) shows a simple lens and P' is the image of a point P . The focal length of the lens is λ. If the coordinates of the point P are x,y,z and that of the image point are x',y',z' as shown in the figure, then we get

$$x' = \frac{1}{1-\frac{z}{\lambda}}\, x \;, \qquad y' = \frac{1}{1-\frac{z}{\lambda}}\, y \quad \text{and} \quad z' = \frac{1}{1-\frac{z}{\lambda}}\, z \qquad ..(3.2)$$

This can be written in the matrix equation form as

$$\{C_h\} = [P]\,\{W_h\} \qquad ..(3.3)$$

where $\{W_h\} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$ represents the homogeneous coordinates of the point P,

$\{C_h\} = \begin{bmatrix} kx' \\ ky' \\ kz' \\ k \end{bmatrix}$ represents the homogeneous coordinates of the image point P',

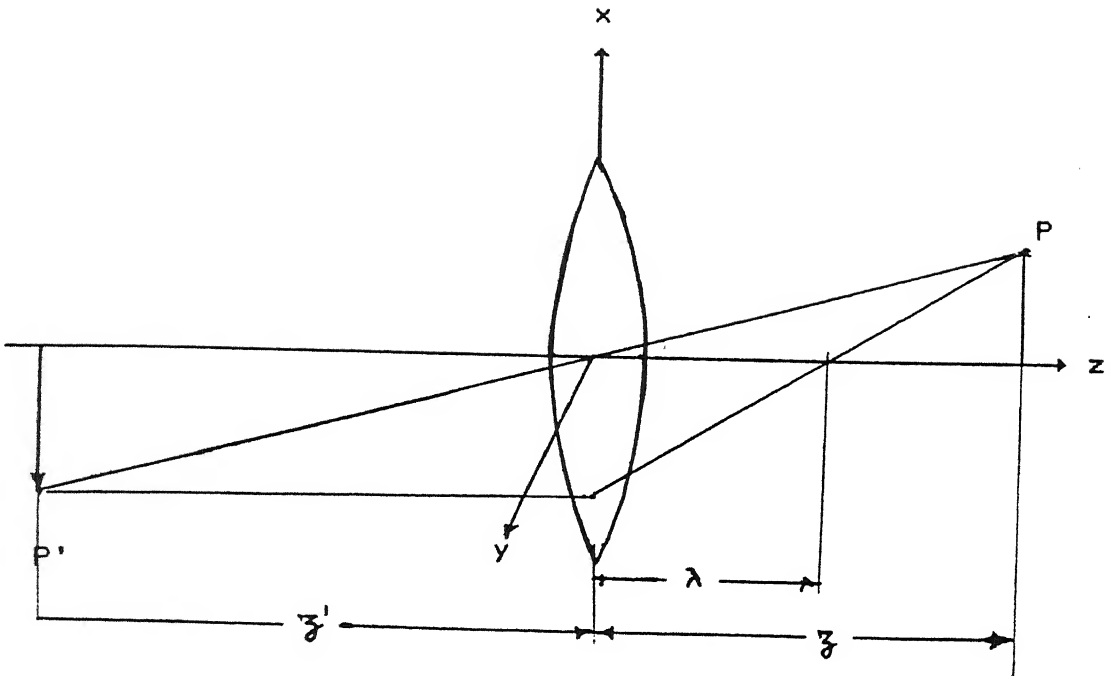


Fig 3.4 (a) Simple lens model

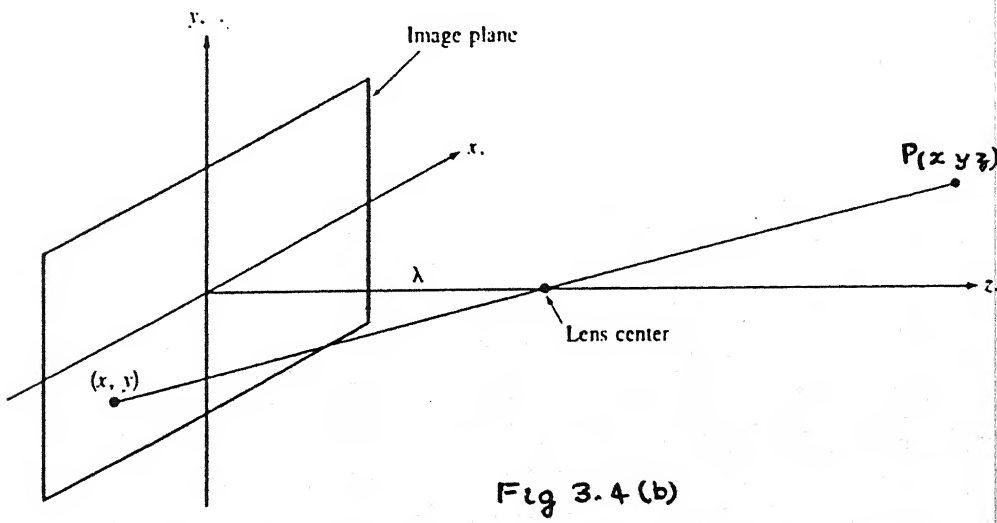


Fig 3.4 (b)

Basic model of the imaging process. The camera coordinate system (x, y,

with $k = \dfrac{1}{1-\frac{\lambda}{z}}$,

and $P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{\lambda} & 1 \end{bmatrix}$ is the Perspective Transformation.

Thus if the image coordinates of a point are known then the coordinates of that point can be obtained from

$$\{W_h\} = [P]^{-1}\{C_h\} \qquad .. \ (3.4)$$

where, $[P]^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{\lambda} & 1 \end{bmatrix}$,

$$\{C_h\} = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} ,$$

and $\{W_h\} = \begin{bmatrix} k_1 x \\ k_1 y \\ k_1 z \\ k_1 \end{bmatrix}$ So that $k_1 = \dfrac{1}{1+\frac{\lambda}{z}}$.

However the depth information z' from the image is not of much use because the camera will have a finite depth of field and hence the objects at different distances will be imaged sharply at

the same z'. Instead, it is assumed that z value is obtained through other means (e.g. by ranging sensor or stereo vision) and then z' is calculated from the last equation of (3.2). In other words, the simple lens has been modeled as a pin hole camera, with the pin hole at the lens centre, as shown in Fig 3.4(b) and we can still use (3.4) where the depth information about z has to be supplied separately.

(iii) Transformation between Camera coordinates and world coordinates :

Let us consider Fig 3.5 which shows a world coordinate system (X,Y,Z) and a camera coordinate system (x,y,z). The camera is mounted on a gimbal which allows pan through an angle θ and tilt through an angle α. The pan angle θ is defined as the angle between x and X axes, and tilt, α is defined as angle between z and Z axes. The offset of the center of the gimbal from the origin of the world coordinate system is denoted by vector $\vec{W}_o$, and the offset of the center of the imaging plane with respect to the gimbal center is denoted by a vector $\vec{r}$.

Suppose that, initially the camera was in *normal* position, in the sense that the gimbal center G and origin of the image plane were at the origin of the world coordinate system, xyz was coincident with XYZ. Starting from *normal* position, the geometrical arrangement in Fig 3.5 can be achieved in a number of ways. We assume the following sequence of steps.

(1) Translate the gimbal centre G (and hence xyz) by the displacement vector $\vec{W}_o$, where X_o, Y_o, Z_o are the components of $\vec{W}_o$ in WCS.

(2) Pan the x-axis by rotating current xyz about the vertical axis by an angle θ.

(3) Tilt the z-axis by rotating current xyz about the current

x-axis by an angle α.

(4) Translate the new xyz axis system by displacement vector $\vec{r}$ where r_1, r_2, r_3 are the components of $\vec{r}$ in XYZ system.

The components of a world point $\vec{W}$ (Fig 3.5) can either be expressed as homogeneous coordinates in XYZ or xyz system. These components are related by,

$$\text{or } \left\{ W_h \right\}_{XYZ} = \left[G\right]\left[R_\theta\right]\left[R_\alpha\right]\left[B\right]\left\{ W_h \right\}_{xyz} \qquad ..(3.5)$$

$$\text{where, } \left[G \right] = \begin{bmatrix} 1 & 0 & 0 & X_o \\ 0 & 1 & 0 & Y_o \\ 0 & 0 & 1 & Z_o \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \left[R_\theta \right] = \begin{bmatrix} C\theta & -S\theta & 0 & 0 \\ S\theta & C\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\left[R_\alpha\right] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha & -S\alpha & 0 \\ 0 & S\alpha & C\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \left[B \right] = \begin{bmatrix} 1 & 0 & 0 & r_1 \\ 0 & 1 & 0 & r_2 \\ 0 & 0 & 1 & r_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Moreover, if $\left\{ C_h \right\}_{xyz}$ are the homogeneous coordinates of the image point then combining 3.4 and 3.5 we can write

$$\left\{ W_h \right\}_{XYZ} = \left[G\right]\left[R_\theta\right]\left[R_\alpha\right]\left[B\right]\left[P\right]^{-1}\left\{ C_h \right\}_{xyz} \qquad ..(3.6)$$

$$\left\{ W_h \right\}_{XYZ} = \begin{bmatrix} k_2 X \\ k_2 Y \\ k_2 Z \\ k_2 \end{bmatrix} \qquad \left\{ C_h \right\}_{xyz} = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix}$$

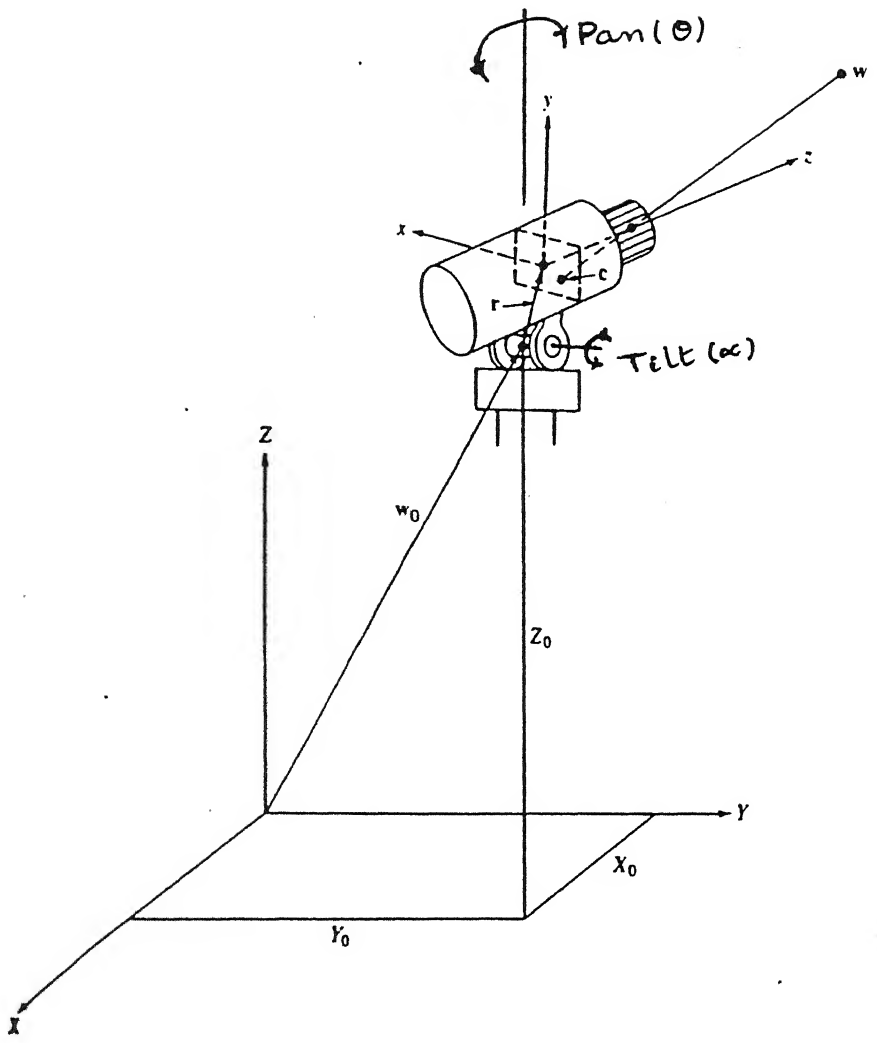
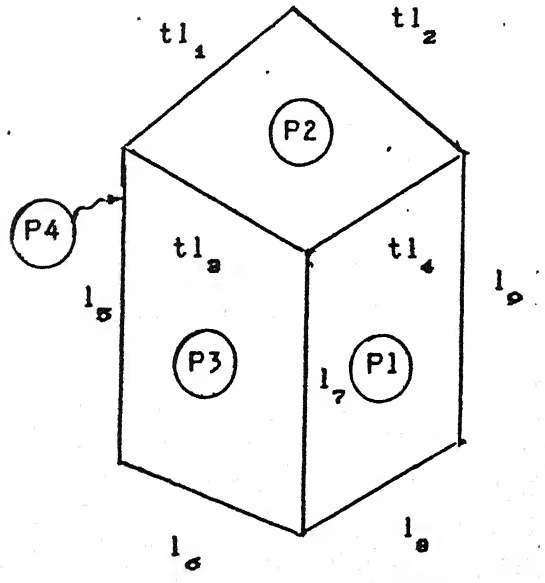


Fig 3.5 Imaging geometry with two coordinate systems.



P- Plane
l- line
tl - top line

Fig 3.6

The three non-trivial equations obtained by term wise matching (of right and left sides of 3.6) and solving for X,Y,Z are

$$Z = \left[\frac{\frac{z}{\lambda}\left\{r_2 S\alpha + C\alpha(\lambda + r_3) + Z_o\right\} + (r_2 + y)S\alpha + r_3 C\alpha + Z_o}{(\frac{z}{\lambda} + 1)} \right] \qquad ..(3.7\ a)$$

Knowing Z we can calculate z by the following relation.

$$z = \lambda \left[\frac{(y + r_2)S\alpha + r_3 C\alpha + Z_o - Z}{Z - r_2 S\alpha - C\alpha(\lambda + r_3) - Z_o} \right] \qquad ..(3.7\ b)$$

$$X = \left[\frac{x*C\theta - y*C\alpha S\theta + \frac{z}{\lambda}\left\{r_1 C\theta - r_2 C\alpha S\theta + S\alpha S\theta(\lambda + r_3) + X_o\right\} + r_1 C\theta - r_2 C\alpha S\theta + r_3 S\alpha S\theta + X_o}{(\frac{z}{\lambda} + 1)} \right] \qquad (3.8)$$

$$Y = \left[\frac{x*S\theta + y*C\alpha C\theta + \frac{z}{\lambda}\left\{r_1 S\theta + r_2 C\alpha C\theta - S\alpha C\theta(\lambda + r_3) + Y_o\right\} + r_1 S\theta + r_2 C\alpha C\theta - r_3 S\alpha C\theta + Y_o}{(\frac{z}{\lambda} + 1)} \right] \qquad (3.9$$

One should have the depth information to calculate X, Y and we are supplying it in the following manner. The work table height is known a priori. The height of the object is also known. When the labeling of the line drawing is done, care is taken to see that the lines that form an umbrella to the object are labelled with a prefix *t (meaning top)*. That essentially gives the altitude of all the lines on the top plane or umbrella plane of the object.

Thus in Fig 3.6, we take two lines (e.g. t11 and t14) that lie on the top of the object (hence Z is known) and are members of grippable planes. So, we can calculate the *four* world points

corresponding to these two lines that are helpful in determining the *center of area* and *orientation of planes.*

If t11 has (X1,Y1) (X2,Y2) as its end points
and t14 has (X3,Y3) (X4,Y4) as its end points

$$X_{centre} = (X_1+X_2+X_3+X_4)/4$$
$$Y_{centre} = (Y_1+Y_2+Y_3+Y_4)/4 \qquad (3.10)$$

Then the angle ϕ (Fig 3.7 (a)) can be calculated by

$$\phi = atan \left\{ \frac{Y_2-Y_1}{X_2-X_1} \right\} \qquad (3.11)$$

We are considering only vertical planes so that ϕ suffices for orientation.

Determination of Joint angles for PUMA-560 :

PUMA-560 needs the specification of the world point in the form of (X,Y,Z,O,A,T). X,Y,Z specify the position of a point *centrally located* between the finger tips in world coordinates,and O A T specify the orientation of the hand [26]. X,Y,Z of the centre point are obtained from (3.10). The orientation of the grippable planes is obtained in terms of ϕ from (3.11). This angle ϕ is used to determine the O, A, T angles in the following way.

With O A T the rotation matrix $\left[R \right]$ is given by

$$\begin{bmatrix} COSO-SOSACT & COCT+SOSAST & SOCA \\ SOST+COSACT & SOCT-COSAST & -COCA \\ -CACT & CAST & -SA \end{bmatrix} \qquad (3.12)$$

Fig 3.7 (a) depicts the tool frame $X_T Y_T Z_T$ attached to the gripper. Let us consider Fig 3.7 (b) which shows the gripper aligned with the object being gripped.

$$\left[{}^{o}R_{tool}\right] = \begin{bmatrix} -C\phi & -S\phi & 0 \\ -S\phi & C\phi & 0 \\ 0 & 0 & -1 \end{bmatrix} \qquad (3.13)$$

The term wise matching of (3.12) with (3.13) yields

$O-T = \phi+90$

Taking $T=0^{\circ}$ and $A=90^{\circ}$, we get

$$O = \phi + 90$$

Finally the inverse kinematics is done by the in-built subroutines of VAL to get θ_1 to θ_6.

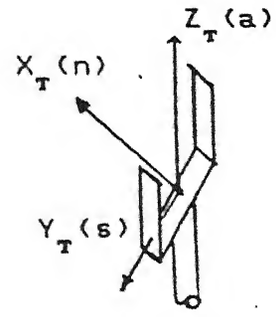


Fig3.7(a)Tool frame definitions

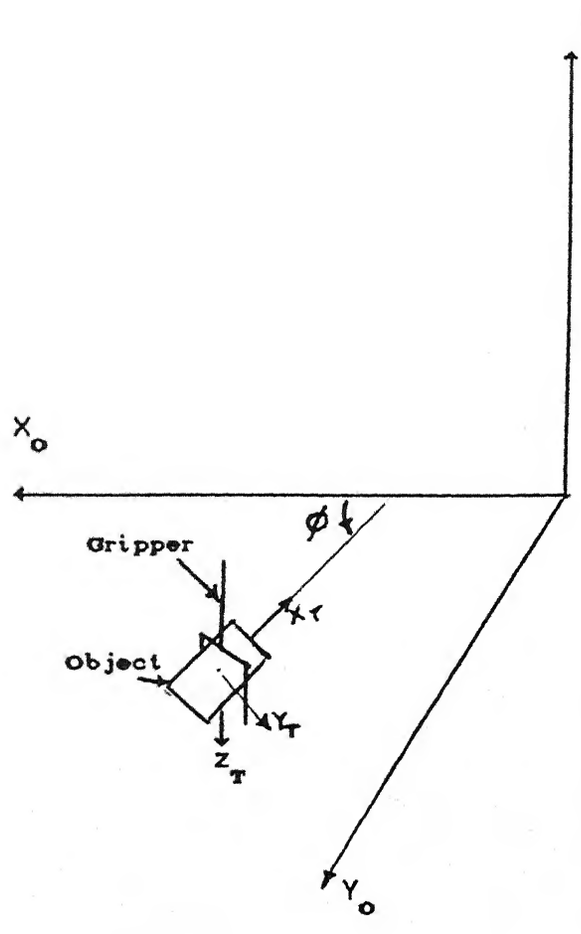


Fig 3.7(b)

o-Base Frame
T-Tool Frame

CHAPTER 4 RESULTS AND DISCUSSION

Introduction :

In this chapter we present the results of implemented algorithms of chapter 2 and 3. First a procedure is discussed to calibrate the camera frame. Then the three objects namely a cube, a hexagonal prism and a triangular prism, are used to illustrate the sequence of operation to determine the grippability. If the objects are found to be grippable then PUMA-560 is used to grip the objects.

Calibration :

The frame grabber or monitor coordinates are shown in Fig 4.1. The camera coordinate system is chosen such that a movement along negative x-axis of camera is shown along positive x-axis on the monitor. Similarly the y-axis of the camera was established . This takes care of the inversion being done by the hardware from the image plane (i.e. the CCD array) to the frame grabber (i.e. as seen in the display monitor).

In chapter 3, we obtained explicit equations (3.7 to 3.9) for determining world coordinates (X,Y) of a image coordinate (x,y) with known Z. The implementation of these equations require knowledge of focal length (λ), camera offsets ($\vec{r}$ and $\vec{W}_o$),angle of pan (θ) and tilt(α). In the present work we have kept camera fixed at same location and orientation. Initially the focal length was set at 45 mm. The camera offset parameters $\vec{r}$ and $\vec{W}_o$ were obtained from direct measurements. These measurements were partly done using PUMA robot. A pointer of known dimension was attached to the end effector and this pointer was positioned at the appropriate locations. The VAL system then gives us the distances. It was found that (see (4.1))

X_o = -662.34 Y_o = -567.97 Z_o = 11.22

r_1 = 61.821 r_2 = -14.41 r_3 = 66.83 (all mm)

The angles θ and α were measured using protractor and were found to be

Pan (θ) = 90^o Tilt(α) = 115^o

Once the transformation parameters were measured, the next step is to verify these parameters. For this, a known world point (i.e. X, Y and Z are known) was imaged to obtain the image point (x,y) and (3.7 to 3.9) were used to verify the results. The following was observed :

$X_{measured}$ = 672.34 mm $Y_{measured}$ = -535.34 mm

$Z_{measured}$ = -548.91 mm

Image coordinates are x = 156 y = 215 (pixel units)

scale in x-direction : 1 pixel = 0.0176 mm

scale in y-direction : 1 pixel = 0.0113 mm

From (3.7 to 3.9) $X_{computed}$ = 560.65 mm $Y_{computed}$ = -460.80 mm

The large errors in computed X and Y is unacceptable and another approach is desired to determine the transformation parameters accurately. It was noticed that (3.7 to 3.9) were sensitive to errors in θ and α than the same percentage errors in the offset parameters. Therefore, these angles θ and α were calculated by trial and error using measured $\vec{W}_o$, $\vec{r}$, $\vec{W}$ and the image point (x,y). This process was repeated for seven known points and the results are presented in Table 4.1 (a). The mean of these angles was taken as the calibrated values for θ and α, and are

X_o = -662.34 r_1 = 61.821

Y_o = -367.97 r_2 = -14.41

Z_o = 11.22 r_3 = 66.83

Sl No	Image Coords		Height	Direct Measurements Using PUMA-560		Pan Tilt (Final)		Focal length
	x	y	Z(mm)	X	Y	α	θ	λ (mm)
1.	141	269	-578.34	713.81	-528.47	87.525	112.875	45.0
2.	50	109	-548.53	737.22	-503.78	87.29	112.843	70.0
3.	111	140	-548.53	715.53	-521.13	87.42	112.869	70.0
4.	156	224	-548.91	672.34	-535.34	87.42	112.8	70.0
5.	157	348	-578.22	672.38	-535.01	87.42	112.826	70.0
6.	257	381	-578.22	660.91	-562.06	87.68	112.698	70.0
7.	257	243	-548.81	660.91	-562.06	87.68	112.78	70.0
					Mean Values	87.4907	112.813	

TABLE 4.1 (a) Calibration

Focal Length λ=40 mm

Sl No	Image Coords		Altitude	X_{calc}	X_{direct}	Y_{calc}	Y_{direct}
	x	y	Z				
1.	165	290	-508.0	536.17	545.97	-514.42	-522.38
2.	243	229	"	589.84	585.38	-557.83	-553.33
3.	317	267	"	552.97	554.47	-595.88	-591.53
4.	246	325	"	504.45	513.75	-555.82	-561.34

TABLE 4.1 (b) Verification

θ = 87.4907 and α = 112.81307

A further check was made to test the correctness of these parameters and the results are presented in Table 4.1 (b). It can be seen that the results are satisfactory.

In our work instead of rotating the camera to the new location the object is rotated to give the desired view. All the calculations to determine the location, orientation etc. are done with reference to the coordinates in the latest view.

Example of a cube :

The digitized image of the cube stored in the *frame grabber* is shown in Fig 4.2. It is a raw image of the object without any pre processing. This is the input for the Low-vision module.

First stage in the Low-vision module is ***edge detection.*** We are using *Sobel's* operator from the ITEX library. The suitability of this edge detected image for further processing is limited by the illumination conditions and the scaling factor, i.e. sobel coefficient chosen (scaling factor would scale the output by dividing the pixel value with this coefficient).

Effect of lighting is illustrated by Fig 4.3 which is the image of a badly lighted object. Fig 4.3 (a) shows the detected edges with a coefficient 45 in sobel operator. One can see clearly that some of the edges are missed out. This can be improved by better lighting conditions. By reducing the coefficient used for Sobel (25) we may be able to get the edge information from the same fig (4.3) but as can be seen in Fig 4.3 (b) there is a lot of noise introduced. This noise would hinder the further processing.

Fig 4.2 is a carefully lighted image and we even accentuated

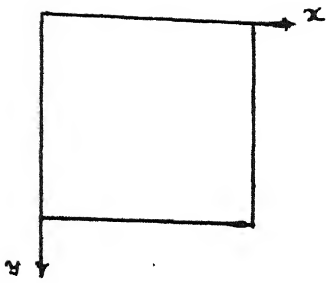


Fig 4.1 Monitor Coordinate System

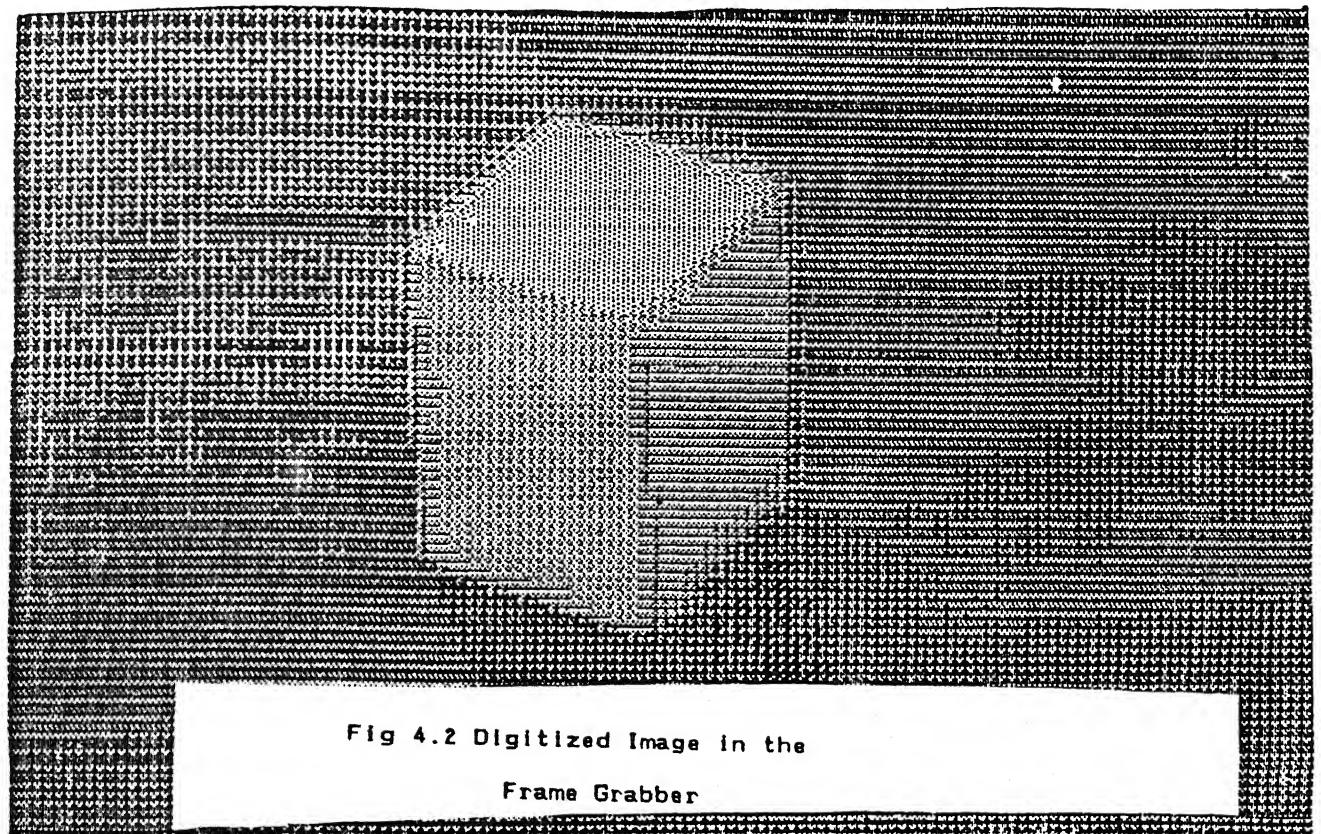


Fig 4.2 Digitized Image in the Frame Grabber

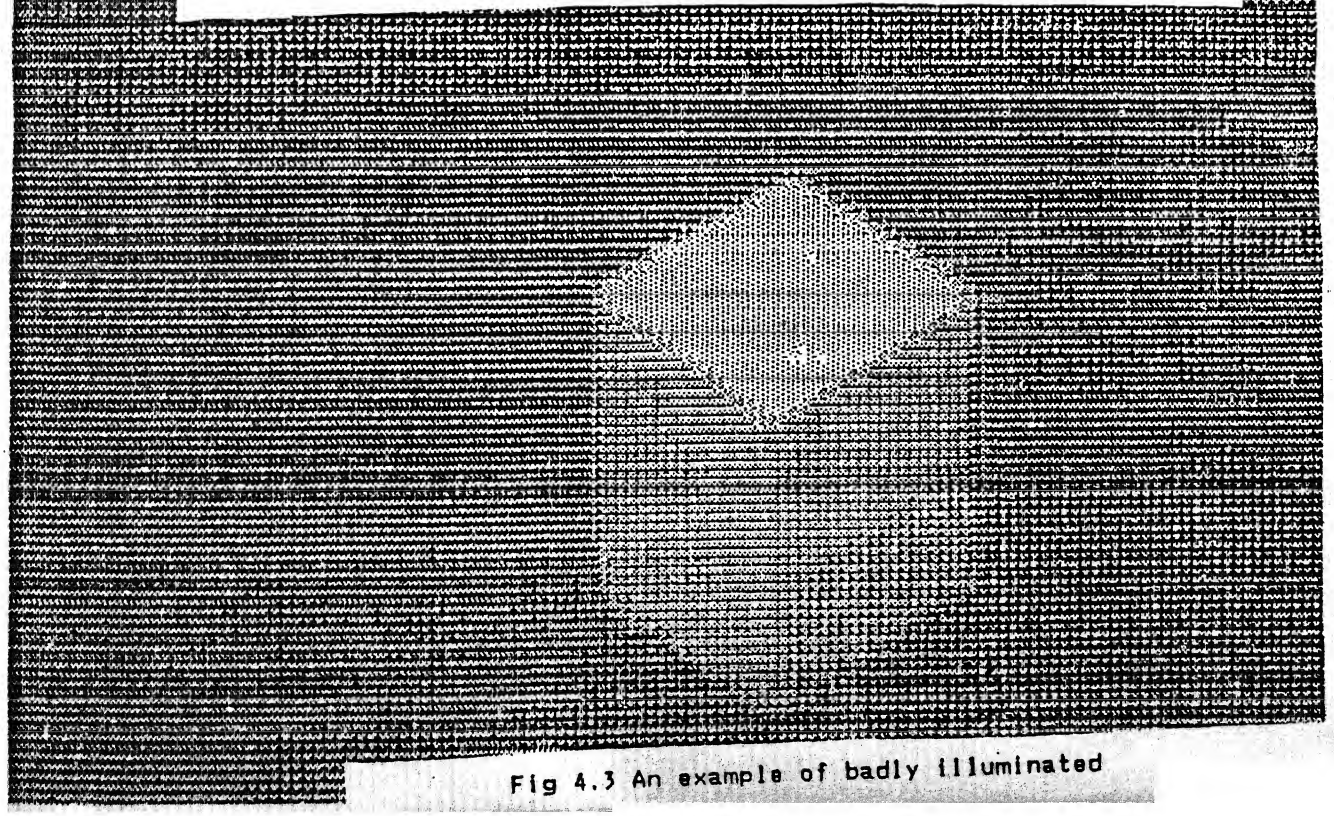


Fig 4.3 An example of badly illuminated

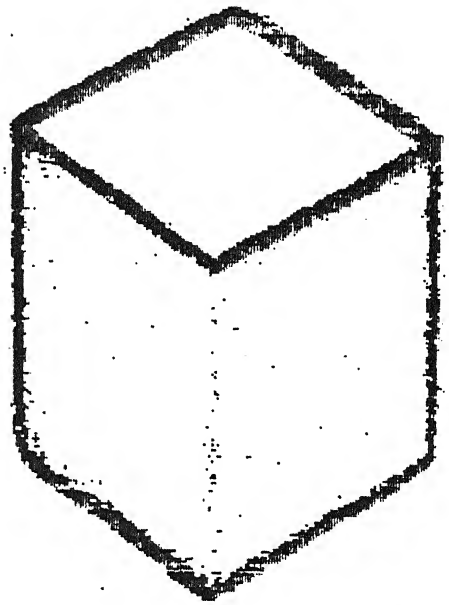


Fig 4.3 (a) Edge Detection performed (on Fig 4.3)

Using a SOBEL 45 operator

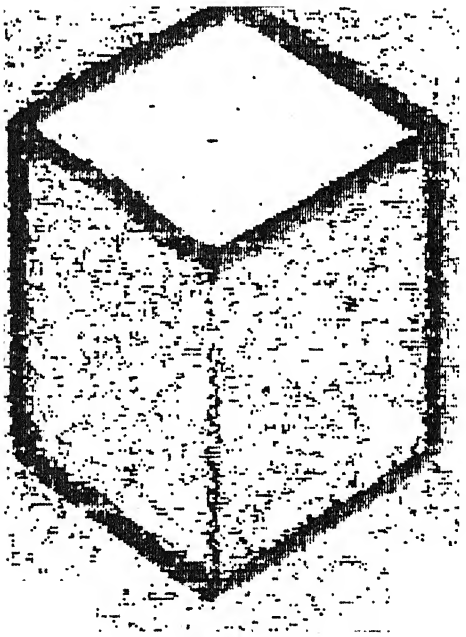


Fig 4.3 (b) Edge Detection performed (on Fig 4.3)

Using a SOBEL 25 operator

some of its edges with a black marker to help us obtain a good line diagram. This image is used for all the subsequent operations described.

Let us see the effect of Sobel's coefficient on the edge information. Fig 4.4(a) shows detected edges with a Sobel coefficient 25. This introduces lot of noise making the further operations difficult. Fig 4.4(b) shows the detected edges with sobel coefficient 45. One can see that all the edge information is present without any undesired noise. Fig 4.4(c) shows the detected edges with sobel 65. One can see that important edge information is lost.

A Sobel coefficient between 35 and 45 was found to work best.

After the edge detection, the edges are thinned to single pixel. Details of the thinning algorithm were presented in chapter 2 . Fig 4.5(a) shows a thinned image of Fig 4.4(a).

Due to thinning some distortions are introduced especially near the corners as evident from Fig 4.5(b) which is the thinned image of Fig 4.4(b). This problem can be handled better if we adopt different processing criteria at blobs. i.e. at all points at which lines either intersect or meet. The thinning strategy we followed fails when a line is perfectly horizontal.

Some of these distortions (especially at the corners) are eliminated by the line finder to a certain extent as it finds the new points of intersections for lines meeting at a point.

The thinned image is broken up into straight line segments. First procedure *SEGMENT*, divides the image into different contours. Procedure *SEPARATE* acts on these contours and breaks them up into valid line segments. At the end of these operations the thinned image transforms into a disjoint line

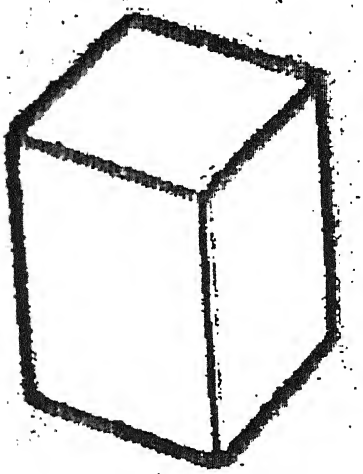


Fig 4.4 (a) Edge Detection performed (on Fig 4.2)

Using a SOBEL 25 operator

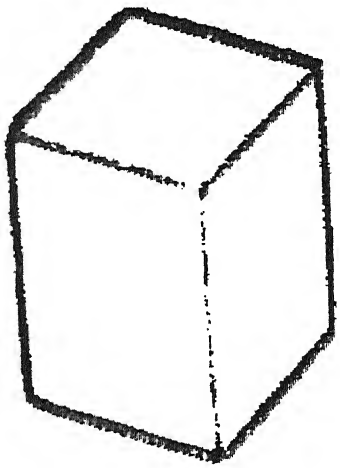


Fig 4.4 (b) Edge Detection performed (on Fig 4.2)

Using a SOBEL 45 operator

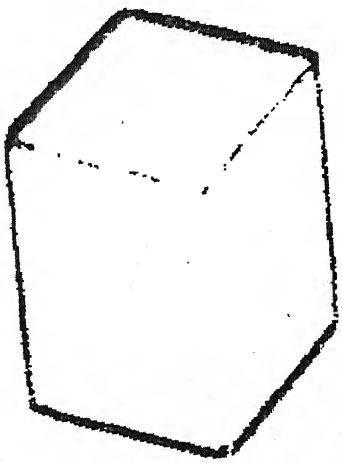


Fig 4.4 (c) Edge Detection performed (on Fig 4.2)

Using a SOBEL 65 operator

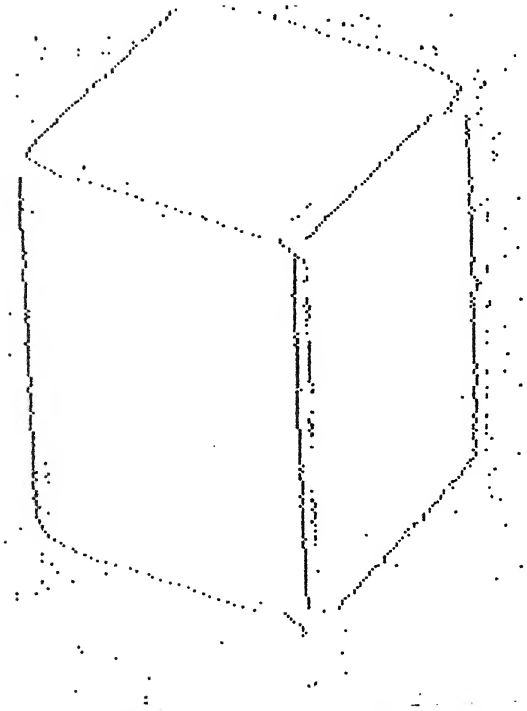


Fig 4.5 (a) Thinning performed on Fig 4.4 (a)

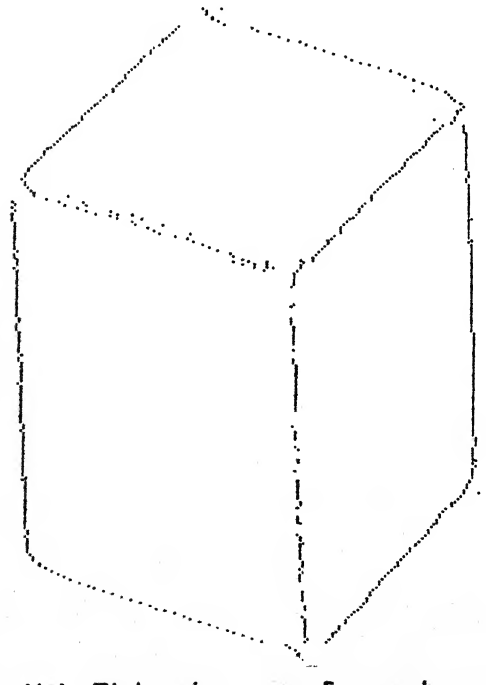


Fig 4.5 (b) Thinning performed on Fig 4.4 (b)

diagram as shown in Fig 4.6(a).

Since the image represents a physical polyhedron, it is essential that the line diagram be continuous. The Hi-vision module can interpret only *doubly connected edge lists*. To achieve this, all lines are extended till they intersect another line. If more than two lines are found to intersect within a specified neighborhood then a common point is assigned, at which all these lines are supposed to converge. Labeling, as explained in chapter 2 is useful in solving the correspondences between the lines. Fig 4.6 (b) shows a connected and labeled line diagram.

Hi-vision module :

Fig 4.7(a) &(b) show the line diagram formed for a cube from 2-different views and the corresponding DCELs. Thus the edge l1 (labeled as t11 as it is a top edge) has a slope of 27.17 and the vertices have coordinates (243 229) and (317 267) in image plane.

Initially all the valid visual planes are formed and also the lines are grouped according to their parallelism. These planes are searched for the existence of grippable planes. When a view is inadequate another view is asked for and similar procedures are followed and the information base is updated.

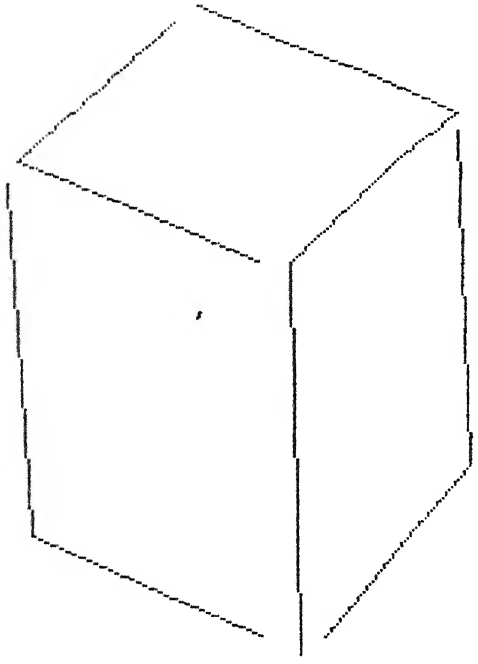


Fig 4.6 (a) Disjoint Line Diagram

[output of Fig 4.5(b) was used]

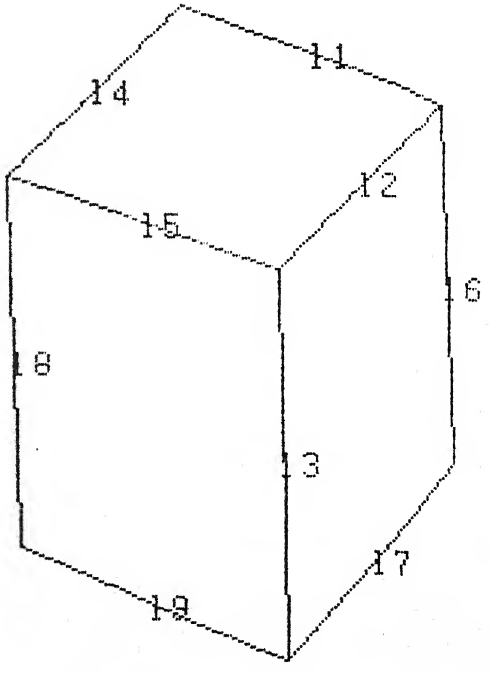
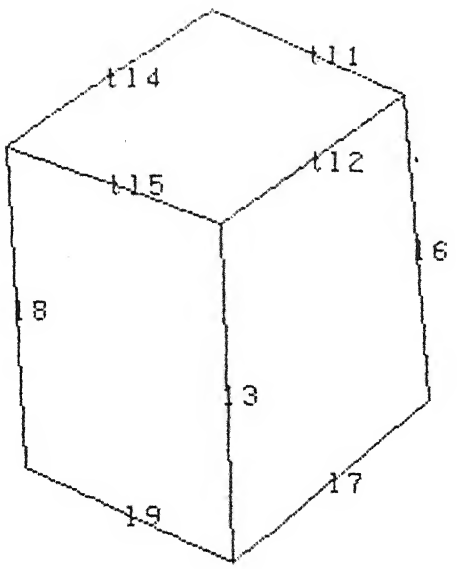


Fig 4.6 (b) Joined and Labeled Line Diagram

Fig 4.7 shows two views of the cube along with the associated DCELs. The execution summary for this example is given below.

```
The valid planes are

((L7 L6 TL2 L3) (L3 L9 L8 TL5) (TL4 TL5 TL2 TL1))

The Group of parallel lines is
((L9 TL1 TL5) (L7 TL2 TL4) (L6 L8 L3))

"Looking for the grippable planes   ... pl. wait"
Another view is required
Now analyzing the SECOND view

The valid planes are
((L12 L8 TL4 L10) (L10 L11 L6 TL1) (TL2 TL1 TL4 TL5))

The Group of parallel lines is
((L11 TL5 TL1) (L12 TL4 TL2) (L8 L6 L10))

The Combined plines is
((L9 L11 TL5 TL1) (L7 L12 TL4 TL2) (L3 L8 L6 L10))
The Combined vplanes is
((TL2 TL1 TL4 TL5) (L10 L11 L6 TL1) (L12 L8 TL4 L10) (L7 L6 TL2 L3)
 (L3 L9 L8 TL5) (TL4 TL5 TL2 TL1))

"Looking for the grippable planes   ... pl. wait"
The grippable planes are
((L10 L11 L6 TL1) (L3 L9 L8 TL5))

(GO TO THE LOCATION 550.5417292501062 -554.3998655055534 -508.16
    132.0386355772563 90.0 0.0)
```

(a) First View (list1)



(b) Second View (list2)

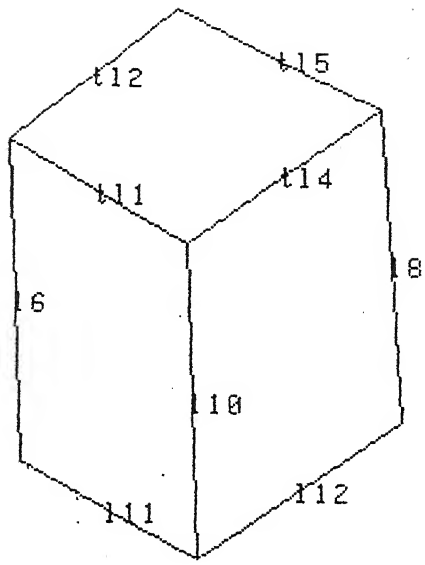


Fig 4.7 Joined and Labeled Line Diagram of a **CUBE**

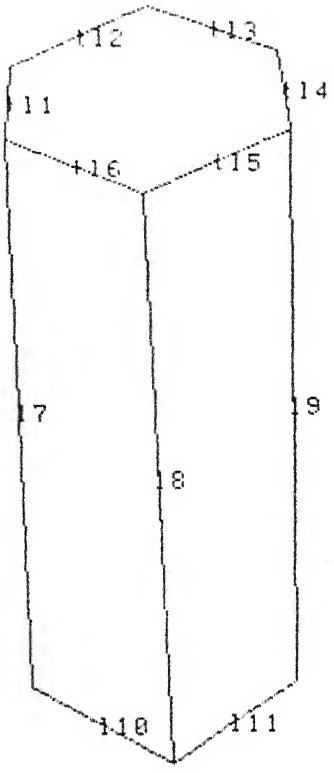
```
(setq list1 '(( t11 27.170 ( 243 229 )( 317 267 ) )( t12 -39.230 ( 317 267 )( 246 325 )
              ( 13 88.080 ( 246 325 )( 251 477 ) )( t14 -38.012 ( 243 229 )( 165 290 ) )
              ( t15 23.360 ( 165 290 )( 246 325 ) )( 16 85.791 ( 317 267 )( 327 404 ) )
              ( 17 -43.829 ( 327 404 )( 251 477 ) )( 18 86.785 ( 165 290 )( 173 434 ) )
              ( 19 28.856 ( 173 434 )( 251 477 ) )))

(setq list2 '(( t15 28.746 ( 236 223 )( 318 268 ) )( t14 -39.274 ( 318 268 )( 241 331 )
              ( 110 87.579 ( 241 331 )( 247 475 ) )( t12 -42.340 ( 236 223 )( 168 285 )
              ( t11 32.204 ( 168 285 )( 241 331 ) )( 18 85.965 ( 318 268 )( 328 411 ) )
              ( 16 87.611 ( 168 285 )( 174 431 ) )( 111 31.067 ( 174 431 )( 247 475 ) )
              ( 112 -38.298 ( 328 411 )( 247 475 ) )))

(setq Z -508.16)
```

(c) Associated DCELs

Fig 4.8 shows two views of the HEXAGONAL PRISM along with the associated DCELs. The execution summary for this example is given below.

```
The valid planes are

((L8 TL6 L7 L10) (L8 TL5 L9 L11) (TL3 TL4 TL5 TL6 TL1 TL2))

The Group of parallel lines is
((L11 TL2 TL5) (TL1 TL4) (L10 TL6 TL3) (L7 L9 L8))

"Looking for the grippable planes   ... pl. wait"

Another view is required
Now analyzing the SECOND view
The valid planes are
((L12 L15 L13 TL3) (L12 L17 L9 L9A TL4) (TL6 TL5 TL4 TL3 TL2 TL1))
The Group of parallel lines is
((L17 TL1 TL4) (TL2 TL5) (L15 TL3 TL6) (L13 L9 L12))

The Combined plines is
((L17 TL1 TL4) (L11 TL2 TL5) (L10 L15 TL3 TL6) (L7 L8 L13 L9 L12))
The Combined vplanes is
((TL6 TL5 TL4 TL3 TL2 TL1) (L12 L17 L9 L9A TL4) (L12 L15 L13 TL3)
 (L8 TL6 L7 L10) (L8 TL5 L9 L11) (TL3 TL4 TL5 TL6 TL1 TL2))

"Looking for the grippable planes   ... pl. wait"
The grippable planes are
((L8 TL6 L7 L10) (L12 L15 L13 TL3))

(GO TO THE LOCATION 568.9401275499559 -572.935738495857 -454.91
    50.68248687905665 90.0 0.0)
```

(a) First View (list1)



(b) Second View (list2)

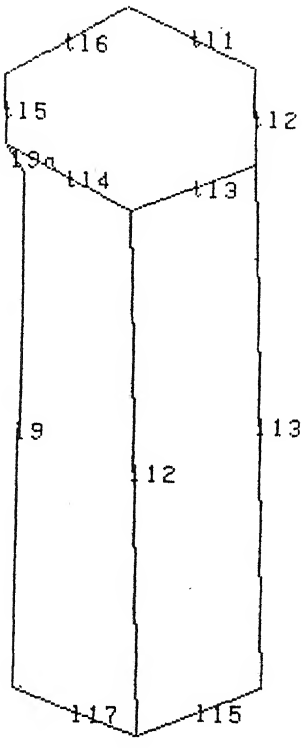


Fig 4.8 Joined and Labeled Line Diagram of a HEXAGONAL PRISM

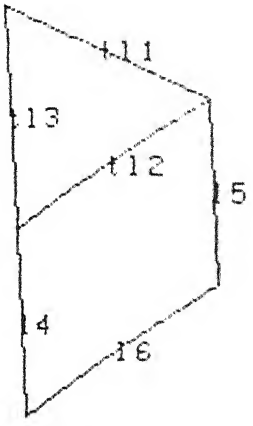
```
(setq list1 '(( t12 -28.463 ( 227 75 )( 168 107 ) )( t11 -86.952 ( 168 107 )( 166 145 ) )
              ( 17 87.554 ( 166 145 )( 178 430 ) )( 110 32.579 ( 178 430 )( 239 469 ) )
              ( t13 20.764 ( 227 75 )( 285 97 ) )( t14 81.641 ( 285 97 )( 291 138 ) )
              ( t15 -28.289 ( 291 138 )( 226 173 ) )( 18 87.450 ( 226 173 )( 239 469 ) )
              ( 19 89.367 ( 291 138 )( 294 426 ) )( 111 -38.004 ( 294 426 )( 239 469 ) )
              ( t16 25.007 ( 166 145 )( 226 173 ) )))

(setq list2 '(( t11 27.313 ( 273 54 )( 333 85 ) )( t12 88.863 ( 333 85 )( 334 137 ) )
              ( t13 -24.953 ( 334 137 )( 276 164 ) )( 112 88.527 ( 276 164 )( 283 443 ) )
              ( t16 -34.367 ( 273 54 )( 216 93 ) )( t15 88.416 ( 216 93 )( 217 130 ) )
              ( 19a 59.012 ( 217 130 )( 226 145 ) )( 19 -89.755 ( 226 145 )( 225 419 ) )
              ( 117 22.470 ( 225 419 )( 283 443 ) )( 113 88.527 ( 334 137 )( 341 416 ) )
              ( t14 29.942 ( 217 130 )( 276 164 ) )( 115 -24.953 ( 341 416 )( 283 443 ) )))

(setq Z -454.91)
```

(c) Associated DCELs

Fig 4.9 shows two views of the triangular prism in *ungrippable configuration* along with the associated DCELs. The execution summary for this example is given below. So, the system keeps asking for another view and after some views the analysis can be aborted. We are aborting it for four views.

```
The valid planes are
((TL3 TL2 TL1) (TL2 L4 L6 L5))

The Group of parallel lines is
((L5 L4) (L6 TL2))

"Looking for the grippable planes   ... pl. wait"

Another view is required
Now analyzing the SECOND view

The valid planes are
((TL3 TL1 TL2) (L5 L8 L7 TL1) (L4 L9 L7 TL3))

The Group of parallel lines is
((TL3 L9) (L5 L7 L4) (TL1 L8))
The Combined plines is
((TL3 L9) (L5 L7 L4) (TL1 L8) (L6 TL2))

The Combined vplanes is
((L4 L9 L7 TL3) (L5 L8 L7 TL1) (TL3 TL1 TL2) (TL3 TL2 TL1)
 (TL2 L4 L6 L5))
"Looking for the grippable planes   ... pl. wait"

Another view is required
THAT 'S HOW THE SYSTEM ASKS FOR ANOTHER VIEW SINCE IT CAN NOT ARRIVE
      AT A DECISION WITH THE AVAILABLE INFORMATION
 We say that the object is UNGRIPPABLE if 4-VIEWS are insufficient to
    arrrive at a decision
```

(a) First View (list1)



(b) Second View (list2)

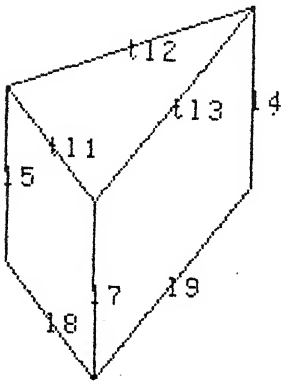


Fig 4.9 Joined and Labeled Line Diagram of a

TRIANGULAR PRISM in *Ungrippable configuration*

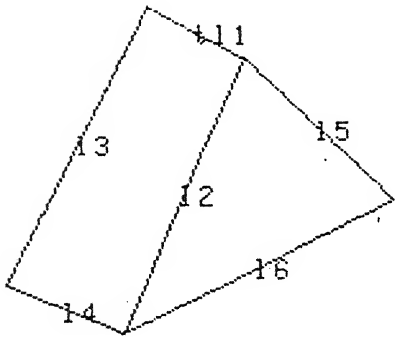
```
(setq list1 '(( t11 29.169 ( 213 256 )( 290 299 ) )( t12 -38.838 ( 290 299 )( 218 357 ) )
              ( 14 87.271 ( 218 357 )( 222 442 ) )( t13 87.131 ( 213 256 )( 218 357 ) )
              ( 15 87.206 ( 290 299 )( 294 382 ) )( 16 -39.790 ( 294 382 )( 222 442 ) )))

(setq list2 '(( t12 -21.597 ( 324 242 )( 223 282 ) )( 15 89.248 ( 223 282 )( 224 362 ) )
              ( 18 55.559 ( 224 362 )( 261 416 ) )( t13 -55.153 ( 324 242 )( 260 334 ) )
              ( 17 89.265 ( 260 334 )( 261 416 ) )( 14 90.000 ( 324 242 )( 324 325 ) )
              ( 19 -55.283 ( 324 325 )( 261 416 ) )( t11 54.545 ( 223 282 )( 260 334 ) )))

(setq Z -527.88)
```

(c) Associated DCELs

Fig 4.10 shows two views of the triangular prism in *grippable configuration* along with the associated DCELs. The execution summary for this example is given below. But since there do not exist two lines that are members of the top plane the location and orientation for PUMA can not be determined. But two parallel planes are determined.

```
The valid planes are

((L6 L5 L2) (TL1 L2 L4 L3))

The Group of parallel lines is
((L3 L2) (L4 TL1))

"Looking for the grippable planes   ... pl. wait"

Another view is required
Now analyzing the SECOND view

The valid planes are
((L9 L3 L8) (TL1 L9 L7 L5))

The Group of parallel lines is
((L5 L9) (L7 TL1))
The Combined plines is
((L5 L9) (L4 L7 TL1) (L3 L2))
The Combined vplanes is
((TL1 L9 L7 L5) (L9 L3 L8) (L6 L5 L2) (TL1 L2 L4 L3))

"Looking for the grippable planes   ... pl. wait"
The grippable planes are
((L9 L3 L8) (L6 L5 L2))
The lines (t) of interest don't exist in the latest case (2nd)
i.e. There are a set of parallel planes but the LOCATION and
ORIENTATION can not be determined because the planes do not have
top lines ( label starting with t ) in them.
```

(a) First View (list1)



(b) Second View (list2)

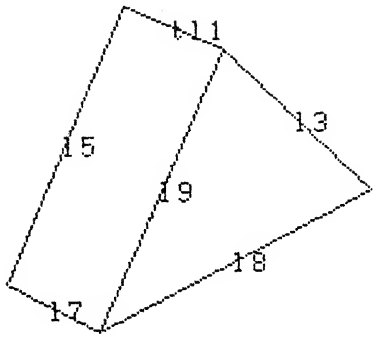


Fig 4.10 Joined and Labeled Line Diagram of a **TRIANGULAR PRISM** in *Grippable configuration*

```
(setq list1 '(( t11 31.172 ( 169 225 )( 207 248 ) )( 12 -69.366 ( 207 248 )( 160 373 ) )
              ( 13 -66.775 ( 169 225 )( 115 351 ) )( 14 26.043 ( 115 351 )( 160 373 ) )
              ( 15 47.796 ( 207 248 )( 265 312 ) )( 16 -30.142 ( 265 312 )( 160 373 ) )))

(setq list2 '(( t11 27.139 ( 182 199 )( 221 219 ) )( 13 47.796 ( 221 219 )( 279 283 ) )
              ( 18 -31.029 ( 279 283 )( 171 348 ) )( 15 -69.268 ( 182 199 )( 134 326 ) )
              ( 17 30.723 ( 134 326 )( 171 348 ) )( 19 -68.786 ( 221 219 )( 171 348 ) )))

(setq Z -529.41)
```

(c) Associated DCELs

CHAPTER 5 CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

For the case of viewing single polyhedron the results have been encouraging. There were certain problems with the formation of the line diagram. But ascertaining the grippability and the actual gripping of the object has been successfully done. There are inaccuracies in determining the position and orientation but we are sure this aspect can be improved. (suggestions are given later in the chapter)

The general approach in this thesis has been to get a solution to a simplified problem with tools tailor made to solve it. We consider only polyhedral objects and have only one object in the scene. We thus avoid solving the very much harder general vision problem. We do not think these methods can be generalized to solve the gripping problem for non-polyhedral objects.

Even so the current approach has some limitations. These can be summarized under three headings :

a) Low-vision : Largely image processing problems

b) Hi-vision : The algorithm which finds the valid visual planes may be improved.

c) Grasp and Pre-grasp : Related to accurately determining the coordinates of the grasp planes and positioning and orienting the gripper correctly.

Starting from the Low-vision module the, important step after getting a good edge detected image is to get the thinned image. As already mentioned in chapter 4 our approach distorts the image at corners and fails for horizontal lines. One might try using a different strategy for blobs and the cases of horizontal lines. But multi-directional thinning and the logical AND OR operations between these images would be too expensive.

The line finder will become faster if all the corners are determined directly rather than doing the segmentation first and then breaking it into lines. Thus one can avoid the rescanning of

the entire 512 X 512 array involved in the segmenting process.

Another major lacuna is that we do not yet have a good theory of how to compute a new viewing location based on the data obtained in the previous views. In the experiments we carried out we were able to get answers by looking at the two views. The second view was arbitrarily chosen to be at 180° rotation of the camera about the vertical about the object. In general this is clearly insufficient and requires a properly articulated theory.

There are large inaccuracies involved in determining the location in WCS from the image plane coordinates. This can be improved by improving the calibration. By improving the camera mount such that the camera offsets and the angles of pan and tilt can be read directly. As of now the position of the camera and the object should remain with in the robot's work envelope because the camera offsets etc. are measured using the robot itself. This resulted in the object being with in the minimum focus distance of the camera. This is one of the causes for large errors. But if the above mentioned mount can be made then proper focusing can be ensured.

The *depth* or range data is not calculated in the current work. This is being fed to the module as explained in Chapter 3. The correspondence problem is being solved interactively by giving the lines absolute labels. This is a difficult problem and will require multiple views taken at small increments. Or this can also be done through a model based approach.

The attached zoom lens (SONY) gives a variable focal length from 17.5 to 105 mm. It currently has no facility for auto-iris, motorized zoom auto-focus etc. which could be of great use if one has to move the camera to a different location for the second view.

REFERENCES

1. L. Rossol, *Computer Vision in Industry - the next decade*, 1983 Robot Vision ed. Alan Pugh pp.11-18.

2. B.K.P. Horn *Robot Vision* MIT Press 1986

3. Holland, Rossol and Ward *CONSIGHT-I: A Vision - controlled Robot systems for Transferring Parts from Belt Conveyors* Computer Vision & Sensor Based Robot ed. Dodd and Rossol pp.81-100 1979.

4. Birk, J.R., Kelley, R.B., and Martins, H.A.S., 1981 *An Orienting Robot for Feeding Work pieces Stored in Bins*.IEEE Trans System Man cybernetics, vol SMC-11, No.2.

5. W.S. Rutkowski and R. Rosenfeld *A Comparison of Corner Detection Techniques for Chain Coded Curves*, technical report No.263, Univ. of Maryland, 1977.

6. O.A. Zuniga and R. Haralick, *Corner Detection Using the Facet model*, in IEEE CVPR 1983.

7. L.Kitchen and A Rosenfeld, *Edge Evaluation Using Local Edge Coherence*, IEEE Trans Systems Man, Cybernetics SMC - 11, No.9, 1981.

8. L.Drechler and H. Nagel, Volumetric model and 3-d *Trajectory of a Moving car Derived from Monocular TV frame sequence of a Street Scene*, in Proceedings IJCAI, 1981, pp.692-697.

9. K. Rangarajan, M.Shah and D Van Brackle *Optimal Corner Detector* CVGIP 48, 1987 (pp. 230 - 245).

10. *A General Photogrammetric Method for Determining Object Position and Orientations* Joseph S.C. YUAN IEEE R & A Vol.5, No.2 April 89.

11. *Determination of Camera location from 2-D to 3-D Line and Point Correspondence* - Y.Liu, T.S. Huang, Oliver D. Faugeras IEEE PAMI 1 Vol.12, No.1 Jan. 90.

12. *Determining Grasp configurations using Photometric Stereo and the PRISM Binocular Stereo Systems* Int. Jr. of Robotics Research, Vol.5, No.1, Spring 1986 Ikeuchi, Nishihara, Horn, Sobalvarro, Nagara.

13. Proceeding of FIFTH International Symposium on Robotics Research 1990 Ed Hiura & Arimoto, *3D Feature Extraction from sequence of Range Data* G Succi, G Sandini, E. Grosso, and M.Tistarelli.

14. B K P Horn, *Understanding Image Intensities* Artificial Intelligence, 8 (1977).

15. Ikenchi & B K P Horn, *Numerical Shape from Shading and Occluding Boundaries* Artificial Intelligence, 17 (1981).

16. Allen M.Waxman , Shimon Ullman, *Surface Structure & 3-D Motion from Image Flow Kinematics*, Int. Journal of Robotics Research vol.4 Fall 85.

17. *Computation of Depth* - Faugeras & Heber Int. Journal of Robotics Research Vol 5 Fall 1986.

18. Hanafusa, H. and Asada H. 1977 *Stable Prehension by a Robot Hand with Elastic Fingers.* In Robot motion planning and control ed M. Brady Cambridge: MIT Press, PP. 323-336.

19. Okada, T. 1979 *Object Handling System for Manual Industry* IEEE Trans. Sys. Man SMC - 9(2): 79-89.

20. Okada, T. 1982 *Computer control of Multi jointed Finger System for Precise Handling* IEEE Trans sys. Man. cyber SMC - 12 (3): 289-299.

21. Salisbury, J.K. and Craig JJ 1982, *Articulated Hands: Force Control and Kinematic issues* Int Journal of Robotics Research,1982

22. Kerr & Roth, *Analysis of Multifingered Hands* Int Journal of Robotics Research Vol 4 No. 4 Winter 1986.

23. Randy C Brost 1988, *Automated Grasp Planning in the Presence of Uncertainity*. Int Journal of Robotics Research Vol 7, No 1, Feb 88

24. Lozano-Perez T. 1976 *The Design of a Mechanical Assembly system* AI-TR-397 MIT,AI lab.

25. Lozano-Perez T. 1981 *Automatic Planning of Manipulator Transfer Movements* Sys Man Cybernetics, IEEE SMC-11, 681-698

26. J Cardillo and MA Sid-Ahmed, *3-D Robot Vision Metrology & and Camera Calibration from Focus Information* VISION 89 Conference proceedings April 24-27, 1989.

27. Fu, Gonzalez & Lee, 1987, Mc Grawhill *Robotics Control, Sensing, Vision and Intelligence.*

28. User's Guide to Val (Programming Manual) 1987, Kawasaki

29. PCVISIONplus Frame Grabber User's Manual 1987, Imaging Technology Incorporated

30. ITEX PCplus Programmer's Manual 1987, Imaging Technology

APPENDIX A

Model PULNiX TM-560 *miniature CCD Camera*

SPECIFICATIONS

Imager		
	Pixel H x V)	500 x 582
	Optical Black	H: 2(F) + 30(R) V: 14(F)
	Image Size(mm)	8.8(H)x6.6(V)
	Dynamic Range	67 dB at 25°C
	Other	Low noise,blooming/smearing suppression
Scanning		625 lines (H) 50 Hz (V) 2:1 Interlace
	Clock Frequency	28.375 MHz
	Pixel Scan Frequency	9.458 MHz
	Horizontal Frequency	15.725 KHz
	Vertical Frequency	50.00 Hz
Sync		Auto Switch
	External Sync	Negative Going pulse
	Hd	15.625 KHz (±5%)
	Vd	50.00 Hz (± 5%)
	TTL level	(+5V to 0V) or
	C-MOS level	(0V to -5V)

Input impedance	4.7 K Ohm
Interlace or Non-interlace	

TV Resolution

Horizontal	370 lines (450 H*) 420 lines 250 lines (non-interface)
S/N Ratio	50 dB type. (56dB with AGC off)
Sensitivity	50 lux (AGC off; F = 1.4)
Minimum illumination	2 lux (AGC on; F = 1.4) 0.6 lux type. (Gain Max; F = 1.4)
Video Output	1.0V p-p composite Video; Sync Negative (RS-170)
AGC	12dB (On); 0dB (Off) Internal Jumper
Gamma	Gamma = 0.45 (Standard) Pot adjustable to = 1
Power Requirements	12V DC ± 10%; Current Consumption: 300 mA
Lens Mount	Mini-Bayonet (C-mount with C-mount adaptor
Auto-Iris	6-pin connector for PULNIX mini auto-iris lenses or standard EE lenses

Temperature Range	$-10^{o}C$ to $+ 50^{o}C$
Camera Mount	Direct 1/4-20 mount hole on bottom of camera
Size (mm)	42 (W) x 32 (H) x 119.5 (L)
(inches)	1 5/8 (W) x 1 1/4 (H) x 4 3/4 (L)
Weight	200 g (7 oz)
Vibration	7 G (11 Hz to 200 Hz)
Shock	60 G shock

APPENDIX B

Model **PCVISION plus** *frame grabber*

SPECIFICATIONS

Video Input

- RS-170/330 or 50Hz CCIR (European), NTSC luminance only
- Two software-selectable video inputs
- DC restoration corrects input signal drift
- Eight 256 x 8-bit Input Look-Up Table allow pre-processing
- Monotonic flash A/D Converter digitizes video at 10 MHz for 512 x 480 (512 x 512) image, or 12.5 MHz for 640 x 480 image (square pixels)
- Anti-aliasing filter provides - 3dB attenuation at 4.2 MHz and -12dB attenuation at 8 MHz
- Signal-to-noise ratio greater than 40 dB
- Programmable gain .67 to 1.33, and offset voltage -0.1V to + 1V in 100 increments each
- Programmed gain and offset voltage values are stored in a nonvolatile memory

Frame Memory

- Frame memory size - 1024 x 512 pixels with 8 bits per pixel

allows storage of two 512 images or one 640 x 480 image

- Memory-mapped access in 64K blocks
- Simultaneous access of of up to eight pixels
- Transparent computer access to frame memory - does not disturb display
- Hardware zoom of 2:1
- Hardware pan on eight pixel boundaries
- Hardware scroll on two line boundaries
- Bit planes protected independently from CPU access and video acquisition

Video Output

- RS-170 or CCIR RGB video outputs with sync available on green channel
- Output bandwidth: 10 MHz, minimum
- Output voltage: 1V p-p into 75 Ω load
- Eight 256 x 8-bit Look-Up Table per output channel
- Three 8-bit Video Frequency Digital-to-Analog Converters

Timing and Synchronization

- Software-selectable system clock source: crystal or

- Two stage PLL provides stable horizontal lock to noisy sources, such as VCRs

- Automatically switches to crystal-generated timing if external sync is lost

- Composite sync or separate horizontal and vertical sync outputs (TTL) to drive display and inputs

- Pixel jitter typically less than 20 ns

- Special sync bus for synchronizing multiple PCVISION *plus* modules for multiple-board configurations

- Host access time: register access inserts no wait states, memory access inserts 0 to 120 ns of wait states for 512 pixel per line image and 0 to 500 ns for 640 pixel per line image

Host Computer Interface

- IBM Personal Computer AT, XT, and PC, and all 100% compatibles

- 8-bit data bus, 24-bit address bus allows mapping into extended address space

- 16 I/O mapped control registers mappable to any 16-byte boundary

Power Requirements

- 13 watts nominal
- +5V @ 2.20A
- +12V 0.140A
- -12V @ 0.030A

Environmental

- Temperature-0-50 degrees Celsius
- Humidity - 10-90% non-condensing